GBASE®

GBase 8a MPP Cluster

Product manual

www.gbase.cn

GBase 8a MPP Cluster Product manual , GBase Technology Limited

Communication Mode

GBase Technology Limited

East Tower of Putian Innovation Industrial Park, No.22, Kaihua Road, Huayuan Industrial Park, Tianjin High-tech Zone

Tel: 400-013-9696 Email: info @ gbase.cn

CONTENTS

1 Document Introduction

This chapter describes the documentation conventions, document reading conventions, and document acquisition of the GBase 8a MPP Cluster product manual.

Readers


This document is mainly applicable to the following engineers:

- Planning Engineer
- Installation Engineer
- Implementation Engineer
- Technical Support Engineer
- Operation and Maintenance Engineer
- Software Development Engineer
- DBA
- Testing Engineer

product version


The corresponding product and version information of this document is shown in Table **1-1**.

Table -11 Product Name and Version Information

Product	edition
GBase 8a MPP Cluster	V953 version

1.1 Document structure

The GBase 8a MPP Cluster product manual consists of the following parts:

- Description chapter: Provide product description information based on various dimensions;
- Process chapters: provide instructions on operation and maintenance;
- Reference chapter: Provide additional information when reference information is needed during operation.

Table -12 Document Composition

Chapter Type	Chapter Name	content validity	Applicable objects
Descriptive class	Product Description	This chapter provides a detailed description of the GBase 8a MPP Cluster product, including product introduction, product structure introduction, deployment plan, enterprise level enhanced features introduction, technical indicators, and other information.	● All readers
Process class	Software installation	This chapter introduces the deployment plan, software installation, uninstallation, upgrade, rollback, and other specific operation processes of GBase 8a MPP Cluster.	● Installation Engineer
	Administrator's Guide	This chapter introduces how to manage clusters. Provide system administrators with service management, cluster management, component and tool management, node management, alarm management, audit management, backup recovery and log management, security management, resource management, cluster performance optimization, high-risk operations, etc.	● Operation and Maintenance Engineer
	Database Management Guide	This chapter provides guidance on the daily management and maintenance of the GBase 8a MPP Cluster database.	● Operation and Maintenance Engineer
Reference Class	appendix	This chapter provides the basic concepts, internal parameter lists, log related information, and error code information involved	● Planning Engineer ● Installation Engineer

Chapter Type	Chapter Name	content validity	Applicable objects
		in the entire usage phase of the GBase 8a MPP Cluster product.	● Operation and Maintenance Engineer ● Software Development Engineer

1.2 Document Reading Conventions

Symbolic conventions

The following symbols may appear in this article, and their meanings are as follows:

Table -13 Symbol conventions

Symbol	explain
 **warning**	Indicates a serious potential danger that, if not avoided, may result in unpredictable consequences such as equipment damage and data loss.
 **be careful**	Indicating potential risks, if these texts are ignored, it may lead to product errors, reduced device performance, or unpredictable results.
 **explain**	Representations are additional information to the main text, emphasizing and supplementing it.

General format conventions

Table -14 General Format Conventions

format	explain
Song typeface	The main text is represented in Song typeface.
bold	The table number and title, image title, attention, and description should be bolded in Song typeface.
Times New Roman	The English language in the main text is represented by Times New Roman.
Uppercase English (such as SELECT)	Represents the GBase 8a MPP Cluster keyword.
italic	Represents variable values that require user customization in syntax
…	Indicates omitted content.

Command Line Format Conventions

Table -15 Command Line Format Conventions

format	significance
bold	Executed SQL statements and operating system commands are represented in bold font.
<>	The parts enclosed by "<>" are mandatory during command configuration.
[]	The parts enclosed by '[]' are optional during command configuration.
#	Behavior comment lines starting with '#'.

Graphical Interface Element Reference Conventions

Table -16 Graphical Interface Element Reference Conventions

format	significance
 Box	Indicates areas that require attention.
 ellipse	Represents a mouse click.
->	Multi level menus are separated by '->'. If you choose "Create Cluster ->Create Collection Center ->Add Server ->Create/Modify User", it means selecting the "Create/Modify User" menu item under the "Create Collection Center" submenu under the "Add Server" menu under the "Create Cluster" menu.

1.3 Obtain and update documents

You can obtain documents through the following methods:

- Apply for documents through sales, pre-sales, or after-sales personnel at Nanjing University General Motors.
- Download the document by logging into the official website of Nanjing University General Motors:
  http://www.gbase.cn/tech_info/473.html

1.4 Feedback

Nanda General Motors welcomes and cherishes your opinions and suggestions. Please provide feedback on this manual through the following methods.

- Telephone feedback:

  Tel: 400-013-9696

- Network feedback:

  Official website: http://www.gbase.cn

  GBase 8a Technology Community: http://www.gbase8a.com

2 Product Overview

This chapter provides a detailed description of the GBase 8a MPP Cluster product, including product introduction, product architecture, deployment plan, enterprise level enhancement features, technical indicators, and other information.

2.1 Product Introduction

2.1.1 product positioning

The GBase 8a MPP Cluster independently developed by Nanjing University General Motors is a mature analytical MPP database in the era of big data. The latest GBase 8a MPP Cluster V9 version of virtual clusters is suitable for system planning and construction of multiple clusters, which can achieve independent planning and unified management of various cluster businesses; Virtual clusters include data management clusters, user management clusters, and cluster version management clusters; Transparent data migration, data association and data sharing can be realized between each logical subset group.

2.1.2 Application Scenario

GBase 8a MPP Cluster is suitable for big data platforms, comprehensive BI systems, data warehouses, and market systems that involve relatively independent business domains or different types of analysis. Different application scenarios run in independent logical sub clusters, and each logical sub cluster is managed uniformly. This not only solves the high cost problem of managing, monitoring, and maintaining multiple physical clusters, but also meets the differentiated characteristics of different business scenarios, Maximizing the utilization of resources enhances the scalability and maintenance capabilities of the cluster.

2.1.3 Technical characteristics

GBase 8a MPP Cluster has diversified platform options, logical architecture that keeps pace with the times, efficient storage of massive data, high-speed loading of massive data, high-performance analysis of massive data, elastic server resource scaling, perfect system resource management, multi-level high availability, data disaster recovery across data centers, convenient data migration, reliable data security Complete SQL standard support and simple and convenient daily operation and maintenance technical features. The ability to meet the increasing demand for data analysis, data mining, data backup, and ad hoc queries in various data-intensive industries.

The specific features are as follows:

Diversified platform choices

- Low cost:
  - PC Server that can fully utilize x86 architecture;
  - Support the deployment of cloud platforms and virtual machine environments.
- Localization:
  - Support domestic servers: Huawei Taishan, Sugon Hygon, Inspur K1, Great Wall, etc.;
  - Support domestic CPUs: Shenwei, Loogson, Zhaoxin, Kunpeng, Phytium, etc.;
  - Support for domestic operating systems: Winning the bid for Kylin, Galaxy Kylin, Puhua Linux, Uniontech, OpenEuler, NFS-China, etc.
- Virtualization: Supports virtual machines based on x86 and PowerPC, such as VMware ESXi.
- Cloud services: Support mainstream cloud platforms such as Alibaba Cloud, Tencent Cloud, and Huawei Cloud.

A logical architecture that keeps up with the times

- Federated architecture: Adopting Shared Nothing+MPP architecture.
- Flexible deployment: a two-level deployment structure with dual clusters of computing storage nodes and management nodes is adopted, which has no single point of failure and good scalability.
- Large scale cluster:
  - The management cluster supports the deployment of up to 64 management nodes;
  - A single computing storage cluster supports the deployment of more than 300 data nodes;
  - The entire cluster supports the deployment of over 1000 data nodes.

Efficient storage of massive data

- Massive data scale:
  - The entire cluster can handle more than 15PB of data;
  - A single data node can process over 50TB of data.

- Flexible distribution strategy:
  - Support three data distribution storage strategies: HASH, RANDOM, and REPLICATED;
  - Support storage of no data copies or multiple data copies;
  - Support users to customize distribution strategies according to business scenario requirements.
- Efficient compression:
  - Support instance level, table level, and column level compressed storage;
  - Support different compression algorithms for different data types, and ideally, the compression ratio can reach over 20 times.

Massive data high-speed loading

- High speed: Single node loading performance can reach 100M/s.
- Multiple network transmission protocols: Supports loading using various network protocols such as FTP, SFTP, HTTP, and HTTPS.
- Parallel data loading from multiple data sources:
  - Support direct loading of compressed format data files such as gzip, snappy, and lzo from HDFS;
  - Support real-time data loading based on Kafka as the data source;
  - Supports loading data using Amazon S3 object storage as the data source.

High performance analysis of massive data

- Smart index:
  - High performance, maintenance-free coarse-grained intelligent indexing technology;
  - Low inflation rate: After the intelligent index is established, the inflation rate does not exceed 1%;
  - Quick positioning: Smart indexes contain column based statistical information that can be directly used during data retrieval and positioning, effectively filtering data.
- Massive parallelism:

- Planner based on MPP technology;
- Rule based and cost based optimizers;
- A scheduler based on asynchronous I/O technology.

- High concurrency:
  - Read write separation: supports data loading while querying;
  - In small query concurrency scenarios, it can support throughput of over 35000 per second.
- No single point performance bottleneck: Different data nodes within the cluster have peer-to-peer computing capabilities.
- Data mining analysis: supports data mining analysis in the In Database mode, such as K-Means clustering, logical regression, linear regression algorithm, etc.
- Full text indexing: Supports full single word indexing, ensuring a 100% query recall rate.

Elastic server resource scaling

- Flexible Free Node:
  - Online rapid expansion: When business and data volume increase, resources can be obtained from the Free Nodes list for expansion;
  - Online fast scaling: When business and data decrease, scaling can be performed to release nodes and enter the Free Nodes list;
  - Online replacement of failed nodes: When a cluster node fails, resources can be quickly obtained from the Free Nodes list for replacement.
- Efficient execution performance: Single node expansion performance can reach 50M/s.
- Controllable execution process: The expansion process can be monitored, paused, resumed, and cancelled with flexible monitoring and management.
- Strong business support:
  - Allow data to be added and written during the shrinking process;
  - Allow data to be added and written during the expansion process;
  - During the replacement process of faulty nodes, support the cluster to perform DQL/DML/DDL operations;

- The scaling and replacement of resources do not affect the normal operation of the business.

Comprehensive system resource management

- Multi resource management: Through flexible configuration of resource pools and resource usage plans, it supports the management of key resources and indicators such as CPU, memory, disk space usage, disk I/O, number of concurrent tasks, priority, runtime, and wait timeout.
- Multi tenant: Physical or logical isolation between tenants is achieved through virtual cluster technology.

Multi level high availability

- Cluster level high availability technology:
  - Cluster dual activity:
    - Support quasi real-time data synchronization between two homogeneous clusters;
    - M-S architecture synchronization, main cluster data can be written, and backup cluster data can be queried;
    - Incremental synchronization based on data blocks greatly improves the efficiency of massive data synchronization compared to traditional logical log based synchronization;
    - The mirror cluster of a virtual cluster supports real-time consistency of data, with two mirrored clusters writing to each other simultaneously.
- Node level high availability technology:
  - The scheduling node (GCluster) ensures data consistency during SQL execution through the Failover mechanism;
  - The management node (GCware) ensures the consistency of cluster metadata through virtual synchronization mechanism;
  - The computing node (GNode) ensures the consistency of primary and replica data through automatic synchronization.
- High availability technology for overall cluster failure: Failover persistence mechanism is used to ensure the integrity and consistency of SQL execution data in case of overall network, server and power failure.

- High availability technologies at the core process level: GNode, GCluster, GCware, and other core processes are monitored in real-time and can be recovered in a timely manner after a failure.

Cross data center data disaster recovery

- Support cross center high availability;
- Support site level cluster disaster recovery to ensure uninterrupted business;
- Support full and incremental data backup at the instance and table levels;
- Support full and incremental data recovery at the instance and table levels;
- Support data backup and recovery with Hadoop:
  - Back up the data in the library to Hadoop;
  - Restore the data files in Hadoop to the library.

Convenient data migration

- Support DBLink functionality between homogeneous and heterogeneous database clusters;
- Supports read and write operations of the source database on the target DBlink table.

Reliable data security

- Comprehensive user authentication and permission management: Provide comprehensive user, role, and account control strategies to ensure the security of cluster database access.
- Efficient and transparent data storage encryption:
  - Transparent encryption/decryption: Data is automatically encrypted and decrypted in the background;
  - Lightweight data encryption/decryption: The impact of encryption and decryption load on overall performance is less than 5%;
  - Encryption/decryption for data columns: Encrypt based on the security level of the data field.
- Rich encryption functions: supports multiple encryption functions, such as AES_ ENCRYPT, ENCRYPT, MD5, SHA1, SHA, SHA256, SM4, etc.
- Dynamic data desensitization: supports five data desensitization functions: default

desensitization, random desensitization, custom desensitization, hash desensitization, and designated position desensitization.

- Support non root users to install, deploy, and run.

Complete SQL standardization support

- Supports SQL92 ANSI/ISO, SQL99 standards;
- Supports interfaces such as ODBC, JDBC, ADO.NET, C API, Python API, and TCL API;
- Supports most SQL 2003 OLAP functions.

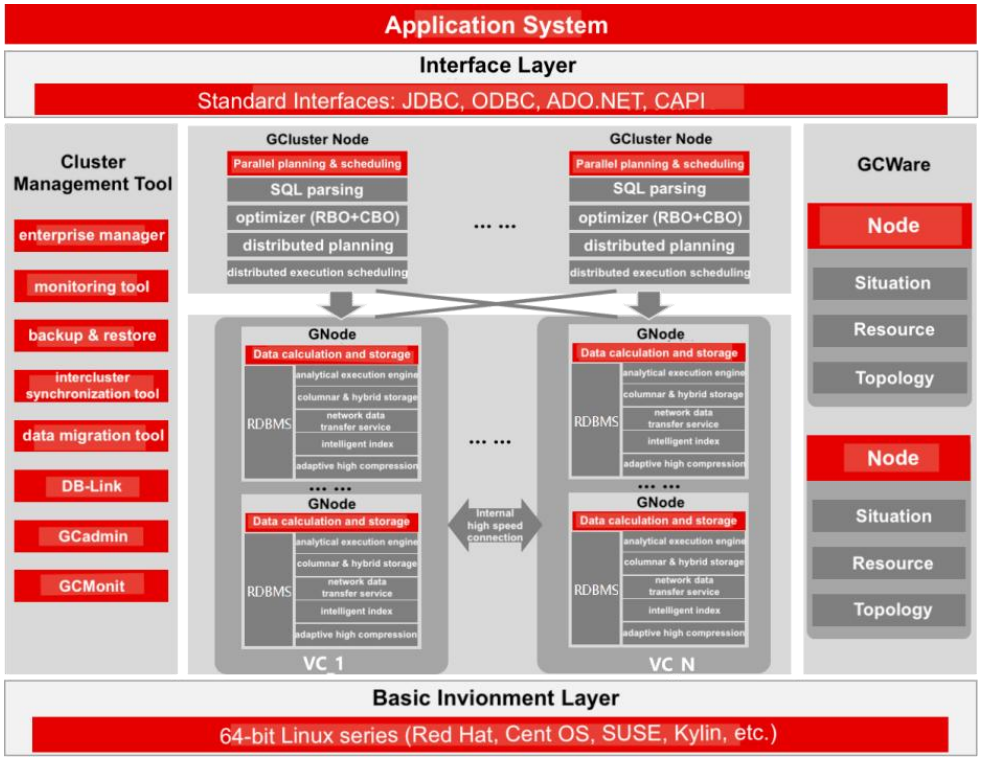
Simple and convenient daily operation and maintenance

- Graphical: Provide graphical management and monitoring tools to simplify administrator management of database clusters;
- Logging: Provides multiple logging functions to facilitate problem tracing.

2.2 Product Architecture

GBase 8a MPP Cluster adopts a distributed federated architecture of MPP+Shared Nothing, where nodes communicate through TCP/IP networks. Each node uses local disks to store data, supporting both symmetric and asymmetric deployments.

Figure -21 GBase 8a MPP Cluster Product Architecture



The GBase 8a MPP Cluster product consists of a total of three core components and auxiliary functional components, including distributed management cluster GCWare, distributed scheduling cluster GCluster, and distributed storage computing cluster GNode. The functions of all components are:

- GCWare

Form a distributed management cluster to provide consistent services for the cluster. Mainly responsible for recording and saving information such as cluster structure, node status, node resource status, parallel control, and distributed queuing locks. When operating on multiple copies of data, record and query operable nodes to provide data consistency status for each node.

- GCluster

Forming a distributed scheduling cluster is the unified entry point for the entire cluster. Mainly responsible for receiving connections from the business side and returning query results to the business side. GCluster will accept SQL, perform parsing optimization, generate distributed execution plans, select actionable nodes to execute distributed scheduling, and provide the results to the business side.

- GNode

Forming a distributed storage cluster is the storage and computing unit of cluster data. Mainly responsible for storing cluster data, receiving and executing SQL issued by GCluster, returning execution results to GCluster, and receiving data from the loading server for data loading.

- GCMonit

It is used to monitor the running status of GCluster and GNode core components in real time. Once the process state of a service program is found to have changed, the corresponding service startup command will be executed according to the content in the configuration file to ensure the normal operation of the service component.

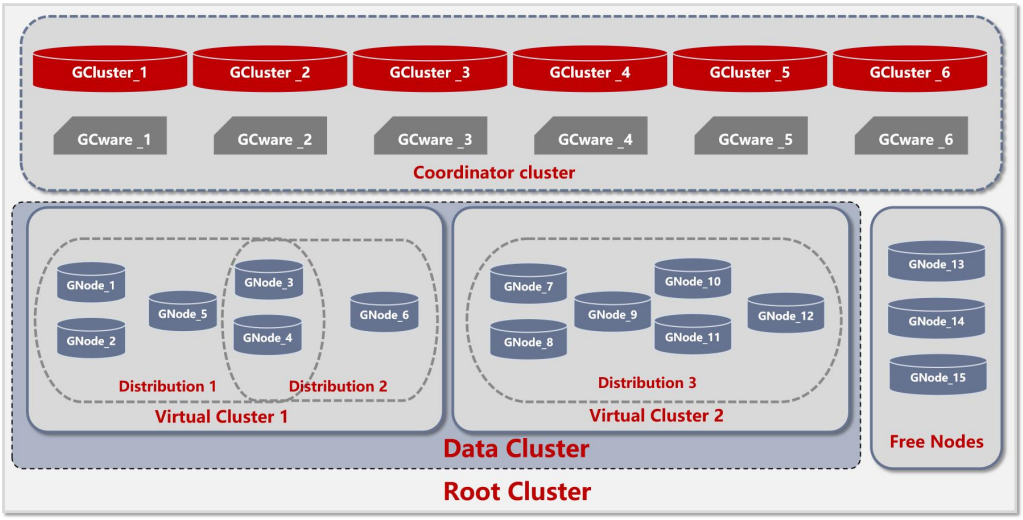
- GCware_ Monit

It is used to monitor the running status of GCware components in real time. Once the service process state changes, it will execute the corresponding service startup command according to the content in the configuration file, so as to ensure the normal operation of the service components.

- GCRecover & GCSyncServer

Used for data synchronization between multiple replicas. Once there is inconsistency in the data files between multiple replicas, the process is called for synchronization to ensure the consistency of the multiple replica data files.

Figure -22 GBase 8a MPP Cluster Product Concept Map



be careful

GCware nodes are recommended to be deployed on GCluster node servers. This composite node that deploys GCluster nodes and GCware nodes together is also known as a coordinator node.

All components mentioned above are divided into logical and virtual concepts, and can be further divided into the following parts:

Logical concept division:

- GCluster Cluster

The distributed scheduling cluster of a cluster is a unified set of entry nodes for the cluster. GCluster Cluster runs gcluster, gcrecover, gcmonit, and gcmmonit services on its nodes.

- GCware Cluster

The distributed management cluster of a cluster is a collection of consistency management nodes in the cluster. Running gcware and gcware on nodes of GCware Cluster_ monit、gcware_ MMONIT service.

- Data Cluster

The distributed data storage computing cluster of a cluster is a collection of data storage computing nodes in the cluster. There are gbased and gc running on the nodes of the Data Cluster_ sync_ Server, gcmonit, gcmmonit services

Virtual concept division:

- VC（Virtual Cluster）

Virtual cluster refers to the division of Data Cluster nodes, with each VC having a fixed number of Data Cluster nodes. The entire cluster is composed of several VCs, all managed by the same set of GCluster Cluster and GCware Cluster, sharing a unified entry point. Different data cluster nodes can be physically isolated according to different business characteristics to form independent VC running independently.

- RC（Root Cluster）

The root cluster is a collection of all GCluster Cluster nodes, GCware Cluster nodes, and Data Cluster nodes, and does not provide services to users. Contains one GCluster Cluster, one GCware Cluster, multiple VCs, and Free Nodes.

2.3 Deployment Plan

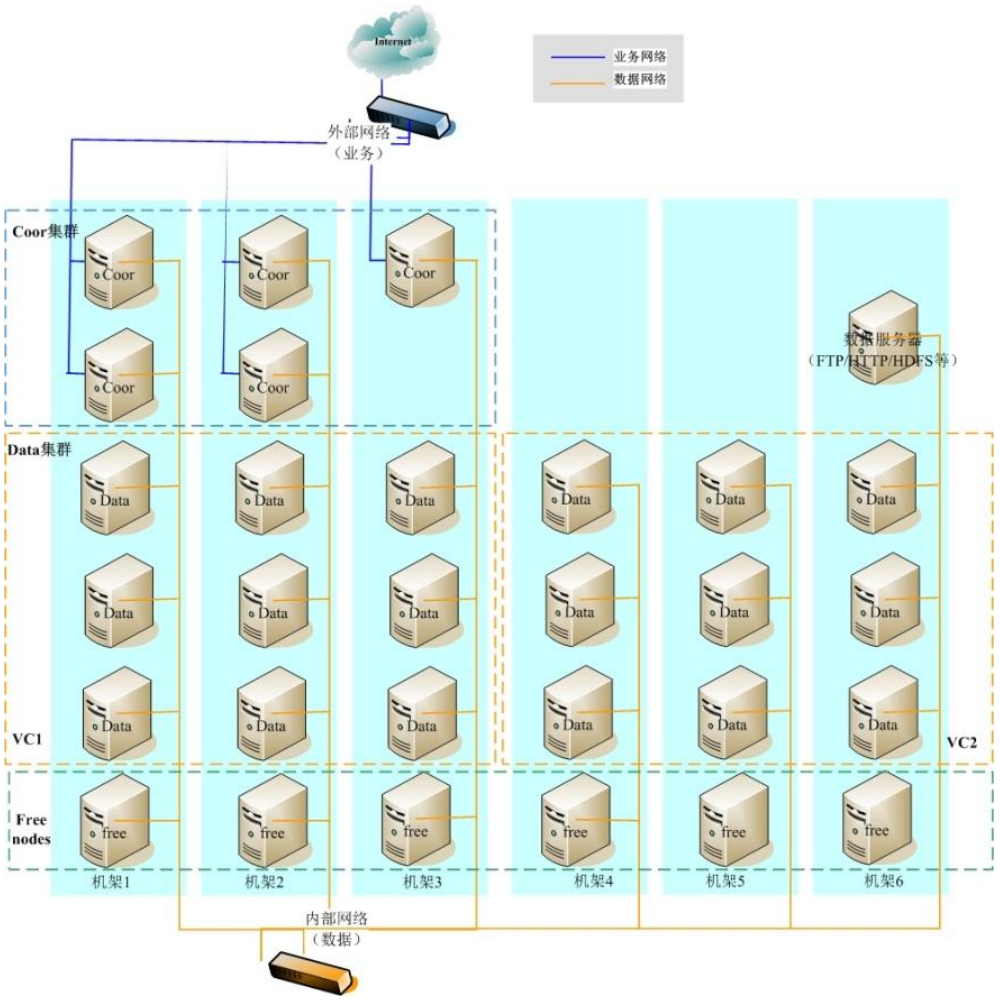
2.3.1 Networking solution

Network Plane Type

The network deployment plan of GBase 8a MPP Cluster can be divided into two physically isolated flat networks, namely the data plane network and the business plane network.

- The data plane network is used for data computation and cluster management within a cluster, also known as the cluster internal network;
- Business plane network is used for business system access, also called cluster external network;
- GBase 8a MPP Cluster is composed of GCluster Cluster, GCware Cluster and Data Cluster. When a two plane network is used for networking, the nodes in GCluster Cluster access the data plane and service plane respectively. You can configure the intranet IP and extranet IP for each node in GCluster Cluster; Each data node in the data cluster only accesses the data plane and only needs to configure the internal network IP; Each node in the GCware Cluster only accesses the data plane and only needs to configure the internal network IP;
- It is recommended to deploy the GCware node and GCcluster node on the same server, and use the same internal network IP for both GCware and GCcluster nodes.

Figure -23 GBase 8a MPP Cluster Network Deployment Plan



Networking solution

It is usually recommended to adopt a two plane network networking scheme, as shown in the network topology diagram -23 of the networking scheme. In the figure, racks 1 to 6 are physical racks. The blue block diagram shows the Coordinator node (a composite node deployed together with GCluster and Gcware nodes), and the orange block diagram shows the Data Cluster node.

Data interaction servers (FTP/HTTP/HDFS, etc.) are deployed on a data plane network.

The internal network of the cluster adopts an Ethernet network or Infiniband network with a bandwidth of 10GE or above, and the internal network aggregation switch adopts a bandwidth of 40GE; The external network of the cluster needs to set the network bandwidth according to the business concurrency, data volume and other requirements. If the big data platform does not involve a large amount of data exchange, it is recommended to use 1GE and above gigabit networks.

Configure the internal network and external network of the cluster for high availability, that is, each IP address of each cluster node is bound with two network interfaces and connected to two switches of each other.

2.3.2 planning scheme

Suggestions for cluster size and node type

Based on the cluster size, it is recommended to allocate the number of Data nodes, GCluster nodes, and GCware nodes in the cluster as follows:

Table -21 GBase8a MPP Cluster Node Type Allocation

Number of Data nodes (replaced by D, unit:)	Number of Gcluster nodes (unit: number)	Number of Gcware nodes (unit: number)
D<10	three	3 or the same number of nodes as Gcluster
10≤D<30	five	3 or the same number of nodes as Gcluster
30≤D<100	7 or 9	3 or the same number of nodes as Gcluster
D≥100	9 or 11	3 or the same number of nodes as Gcluster

Suggestions for cluster deployment plan

- Symmetrical deployment

Deploy GCWare, GCluster, and GNode on the same node, where the same server is both a Coordinator node and a Data node.

- Asymmetric deployment

Deploy GCWare, GCluster, and GNode on different nodes

- Deploy GCluster Cluster, GCware Cluster, and Data Cluster on different servers;
- Deploy GCluster Cluster and GCware Cluster on one server, and Data Cluster on a separate server.

 **be careful**

GCluster Cluster, GCware Cluster, and Data Cluster all require network interoperability and excellent communication quality;

2.3.3 hardware requirements

Hardware configuration requirements

Table -22 Hardware Configuration Requirements

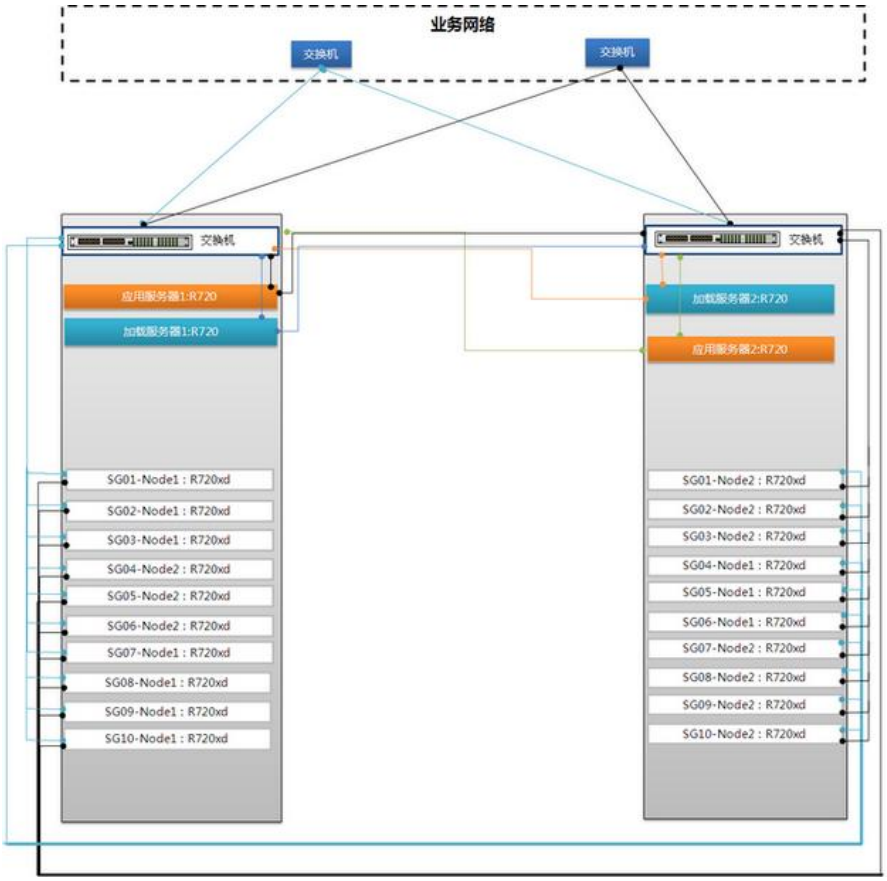
Hardware	to configure
CPU	Minimum configuration: 1 × 2-core 2.0GHz Recommended configuration: above 2 × 8-core 2.0GHz Supports CPU architectures such as X86 and ARM, as well as domestic CPUs such as Kunpeng, Hygon, Loogson, Zhaoxin, and Phytium
Memory	Minimum configuration: 32GB Recommended configuration: 256GB or more
network card	Business plane using 1GE or above Data plane using 10GE or above
Hard drive	Minimum configuration: SATA 7200 rpm 100GB Recommended configuration: SAS 10k rpm or above (SSD, Nvme storage)
Disk RAID configuration	● Management node (GCluster node): The disk where the operating system is located occupies an exclusive RAID group, and the RAID group level is RAID1; The non operating system disk has an exclusive RAID group, and the RAID group level is RAID5. ● Data node: The disk where the operating system is located occupies an exclusive RAID group, and the RAID group level is RAID1; Configure RAID5 on non operating system disks.
disk space	According to the actual needs of users, disk capacity can be flexibly set. ● Management node (GCluster node): Operating system disks ≥ 600GB, and each non operating system disk ≥ 600GB. ● Data nodes: Operating system disks ≥ 600GB, and each non operating system disk ≥ 600GB.

be careful

The memory is less than or equal to 2155M, and the 8a service cannot be started. Use a virtual machine to learn or test 8a, and it is recommended to allocate 2.5GB or more of memory.

Hardware deployment recommendations

Figure -24 GBase 8a MPP Cluster Hardware Deployment Suggestions



It is recommended to have at least three cabinets (with independent power supply for each cabinet), where GBase 8a MPP Cluster product management node servers and data node servers are placed. The network between them communicates through switches. In order to ensure efficient operation of the network, the actual business network in the project also needs to be connected to these backup switches.

The following are the principles for hardware physical deployment and network planning:

- High availability of power supply: The power supply of the cabinet is independent and does not affect each other, following the principle of high availability of host power supply;
- Switch high availability: Each cabinet is equipped with a switch, and the two switches are in a standby relationship. When one switch fails, the other switch immediately provides service, following the principle of switch high availability;
- Node high availability: The 20 hosts marked with dashed lines in the figure are used to deploy GBase 8a MPP Cluster products, following the principle of node high availability.

Configuring Disk RAID

- Configure the local data disk as RAID1 or RAID5, treat multiple physical disks as one large hard disk, and have fault-tolerant and redundant functions.
- GBase 8a MPP Cluster recommends setting the local operating system disk of the

host to RAID1 and the data disk to RAID5 (see 6.15.1.2.1 RAID5 Configuration Reference). RAID5 working mode requires at least 3 identical physical disks.

2.3.4 Software requirements

Operating System Version Requirements

Please ensure that each server has the operating system specified in Table -23 installed. All nodes use the same operating system.

Table -23 GBase 8a MPP Cluster Supported Operating Systems

Operating System Name	edition	framework
Red Hat Linux	6.X、7.X、8.X	X86_ Domestic CPUs such as 64bit and ARM
CentOS	6.X、7.X、8.X	X86_ Domestic CPUs such as 64bit and ARM
SUSE	11 ~12	X86_ Domestic CPUs such as 64bit and ARM
Winning the bid by Kirin	6.0、7.0	X86_ Domestic CPUs such as 64bit and ARM
PowerLinux	7.X	X86_ Domestic CPUs such as 64bit and ARM
Puhua	4.X	X86_ Domestic CPUs such as 64bit and ARM
Galaxy Kylin	V7、V10	X86_ Domestic CPUs such as 64bit and ARM
NFS-China	3~4	X86_ Domestic CPUs such as 64bit and ARM
openEuler	V2 sp5~ sp9	X86_ Domestic CPUs such as 64bit and ARM

be careful

- The operating system versions of all nodes within a Virtual Cluster must be consistent. It is recommended that the operating system versions of all nodes in the entire cluster be consistent.
- Under the Centos 8/Redhat 8 operating system, additional installation of Python 2. X is required, and the default Python program needs to be changed to Python 2. (Including two rpm packages, Python and Python libs)

2.4 Enterprise level enhanced features

2.4.1 Distributed data storage

Column and column mixed storage

Data is organized and physically stored in columns on disk. Faced with the disk I/O bottleneck of massive data analysis, analytical databases store table data in columns, and column storage architecture has natural advantages for queries, statistics, and analysis operations. Its advantages are reflected in the following aspects:

Reduce I/O

Only accessing the columns involved in the query will generate disk I/O. Columns that are not involved in the query do not require access and do not generate disk I/O.

High compression ratio

The compression ratio can reach 2-20 times.

Supports mixed storage of rows and columns

GBase 8a MPP Cluster supports mixed storage of rows and columns. For the cluster architecture of column storage, when the operation involves a large number of columns and the accessed data records are very discrete, it will cause a large amount of discrete I/O. The column column mixed storage function improves disk I/O performance by storing redundant rows of information.

Distributed Storage

GBase 8a MPP Cluster can process structured data above PB level. For large table data, random data storage and distribution strategy mode or hash data storage and distribution strategy mode can be used. Users can choose appropriate data storage distribution strategies according to the needs of business scenarios, in order to achieve the best balance between performance, reliability, and flexibility.

Random data storage distribution strategy pattern

The random data storage distribution strategy pattern refers to the database creating a randomly distributed distribution table, and the data will be randomly and evenly distributed to various data nodes during data warehousing.

Hash Data Storage Distribution Policy Pattern

The hash data storage distribution strategy mode refers to the processing of each data in the original data according to the specified hash distribution column when the data is stored, and the processed data is loaded into a specific hash bucket according to the hash value, with each hash bucket corresponding to a cluster data node. In this way, the data obtained by each node has a common feature (spec

ified columns have the same hash value), and the optimization engine can optimiz e the query plan based on these common features during query to achieve the go al of shortening query time.

Virtual cluster

- A virtual cluster can contain one or more VCs. Each VC is a physical clust er, managed by the same coordinator cluster, and runs independently within t he virtual cluster, sharing a unified entry point; Each virtual cluster (VC) is composed of a set of Data nodes, and all virtual clusters are controlled by a set of Coordinator Clusters. Each virtual cluster operates independently with out affecting each other;
- The unified access portal provided by virtual cluster technology can achieve unified access to warehouses and clusters. For applications, access to multiple physical clusters is transparent, accessing a unified cluster. However, the ori ginal physical cluster can be planned into multiple virtual clusters based on t he business system internally;
- With permission, each virtual cluster can access each other.

Efficient compression

- Efficient transparent compression technology can automatically select the opti mal compression algorithm based on data types and data distribution patterns, minimize the storage space occupied by data, reduce I/O consumption of qu eries, and improve query performance. You can set instance level, table level, and column level compression options to flexibly balance the relationship be tween performance and compression ratio, and the compression and decompre ssion processes are transparent to users;
- Compared to traditional row storage databases, efficient and transparent comp ression technology can bring about an order of magnitude of performance im provement.
  - The compression ratio can reach 2-20 times or even better, far higher th an line storage;
  - Save 50% -95% of storage space, greatly reducing data processing energ y consumption;
  - Built in various compression algorithms of different levels;
  - In the compressed state, the I/O requirements are greatly reduced, and th e data loading and query performance is significantly improved.

Smart Index

- Smart index is a coarse-grained index that generates a data packet for every 65536 rows of data. Each packet automatically establishes a smart index upo n data entry, including filtering and statistical information. Statistical values c

an be obtained without unpacking during data queries, which can further redu ce I/O and significantly optimize complex queries.

- The intelligent index in the table is automatically created, without the ne ed for users to manually establish and maintain it;
- The intelligent index itself takes up very little space and has good scala bility. After establishing the intelligent index, the storage space is almost not inflated;
- The speed of building a smart index is fast, and the speed of building a smart index for subsequent data packets will not be affected by the pre vious data packets.

- Compared to traditional database indexing technology, smart indexes are built on data packets (coarse-grained indexes) and each field is automatically inde xed, while traditional indexes are built on each row of data (fine-grained ind exes). Therefore, accessing smart indexes requires less I/O (tens of thousands of parts) than accessing traditional indexes. Meanwhile, smart indexes occup y approximately 1% of the data space, while traditional database indexes occ upy 20% to 50% of the data space.

Massive parallel computing

- GBase 8a MPP Cluster Single Node Parallel Technology

  GBase 8a MPP Cluster implements automatic and efficient parallel processing technology for data loading and querying, fully utilizing SMP multi-core CP U resources to process massive amounts of data in parallel. At the same tim e, GBase 8a MPP Cluster has intelligent algorithm adaptation function. For e xample, flexible JOIN processing methods support HASH JOIN, NEST-LOOP JOIN, MERGE JOIN, and more. Based on different data distributions and c haracteristics, different algorithms will be intelligently selected for processing. This also fully solves the performance pressure brought by JOIN operations in various industry applications, especially for multi table JOIN operations wi th more than 10 tables.

- The main features of GBase 8a MPP Cluster technology are:
  - Distributed parallel planner, combined with cluster characteristics, perform s distributed processing on operators to generate suitable

    Distributed execution plan;
  - Ensure efficient execution of plans through rule-based and cost based opt imization;
  - The scheduler adopts asynchronous I/O and other technologies to ensure efficient and reliable scheduling.

2.4.2 Workload Management

Cluster interface driver provides load balancing

The cluster interface driver can effectively balance the load of upper layer application requests. The application layer sends requests to the corresponding nodes, which complete SQL parsing and generate execution plans, coordinate the concurrent participation of cluster related nodes in computation and processing, improve the concurrency of the entire cluster, and fully leverage cluster performance.

Multi tenant resource management

GBase 8a MPP Cluster can achieve isolation of physical or logical resources between tenants through virtual cluster technology.

- Resource management within each node under GBase 8a MPP Cluster

  Under GBase 8a MPP Cluster, each node can configure and manage its internal CPU, memory, disk space, and I/O resources. The CPU can control the CPU priority and percentage of controlled SQL usage, as well as manage SQL concurrency and parallelism; Memory can control the upper limit of controlled SQL operator buffer usage; I/O can control the upper limit of the read and write rate of the controlled SQL disk; And control over the use of disk space.

- Overall resource management of GBase 8a MPP Cluster virtual cluster

  Within the virtual cluster, flexible configuration of resource quotas and query priorities for different applications and users is achieved through resource management and resource groups. At the same time, under a comprehensive permission authorization management mechanism, different applications and tasks can run simultaneously in the same cluster while being isolated from each other.

Online expansion

GBase 8a MPP Cluster supports online scaling and scaling of cluster data nodes, resulting in higher execution efficiency and less impact on business. GBase 8a MPP Cluster can expand the computing and storage capacity of the system by adding data nodes, and can flexibly manage and control the status during the expansion process, supporting pauses, restores, cancellations, and so on; The system supports online expansion, with approximately linear performance improvement after expansion, without interrupting the current system operation, and supports expanding multiple nodes at once; It can support multi-level extension methods such as instance level, library level, and table level.

GBase 8a MPP Cluster has online capacity expansion capabilities:

- Online dynamic expansion of cluster nodes;
- Dynamically expanding data nodes online;
- The execution scheduling node and data computing node can be independently expanded as needed.

2.4.3 Data security

data encryption

GBase 8a MPP Cluster has soft encryption function for database landing data, which can meet users' security needs and improve system security. Data encryption implements encryption requirements of different granularity at the table or column level. Data encryption supports the following features:

- Support encrypted keyword encryption for table creation;
- Support encryption requirements of different granularity at the table or column level;
- Support query of table level encryption attributes;
- Support key certificate management.

Data Desensitization

- GBase 8a MPP Cluster has a dynamic data desensitization function. Developers and database administrator can control the exposure of sensitive data, generate desensitized data at the database level, and simplify the security design and coding of the database application layer;
- Dynamic data desensitization does not truly alter the actual data stored in the table, but only applies this feature to control the data returned during queries. Dynamic data desensitization supports five data desensitization functions: default desensitization, random desensitization, custom desensitization partial, hash desensitization sha, and designated desensitization keymask.

2.4.4 Data reliability

Data reliability

- Multi sharding and multi copy mechanism of GBase 8a MPP Cluster

  GBase 8a MPP Cluster ensures the high availability of the cluster through a multi copy redundancy mechanism. The table data in the cluster is divided into multiple shards and stored on different nodes. Each shard can provide redundancy for one or more replica data, and the number of replicas and shards

in the cluster can be flexibly configured. Replicas can be sharded to any node in the cluster, and more master and replica can be allocated to nodes with high host performance and large storage space according to the configuration. Automatic data synchronization will occur between the primary and secondary replicas.

- The multi copy redundancy mechanism can reduce the barrel effect of node failures;
- When a data node fails, the system automatically switches to its replica data node for work, ensuring subsequent business continuity;
- Supports dual active cluster deployment.

- Automatic Switching Mechanism of GBase 8a MPP Cluster

  Node failures are transparent to applications and do not interrupt ongoing business. The load of abnormal servers can be evenly distributed among several normal servers where the replica is located. Once the service of the faulty node returns to normal, the GBase 8a MPP Cluster faulty node will synchronize with other complete replica nodes to recover the node's data. Service will be provided immediately after the recovery is completed, This maximizes the prevention of performance jitter caused by the cask effect after failover.

Backup Recovery

GBase 8a MPP Cluster provides a dedicated backup and recovery tool (gcrcman), allowing users to easily backup and recover data throughout the entire cluster, preventing data loss or damage from adversely affecting user business, and ensuring rapid system recovery in abnormal situations. The backup and recovery tool is automatically installed on $GCLUSTER with the installation of the cluster_ Under the BASE/server/bin directory.

- GBase 8a MPP Cluster provides cluster level, database level, and table level full backup, incremental backup, full recovery, and incremental recovery functions according to user business needs.
- GBase 8a MPP Cluster also supports data backup/recovery with Hadoop, backing up data from the library to Hadoop or restoring data files from Hadoop to the library.

2.4.5 Data loading and integration

2.4.5.1 Data loading

Cluster loading has the following characteristics and advantages:

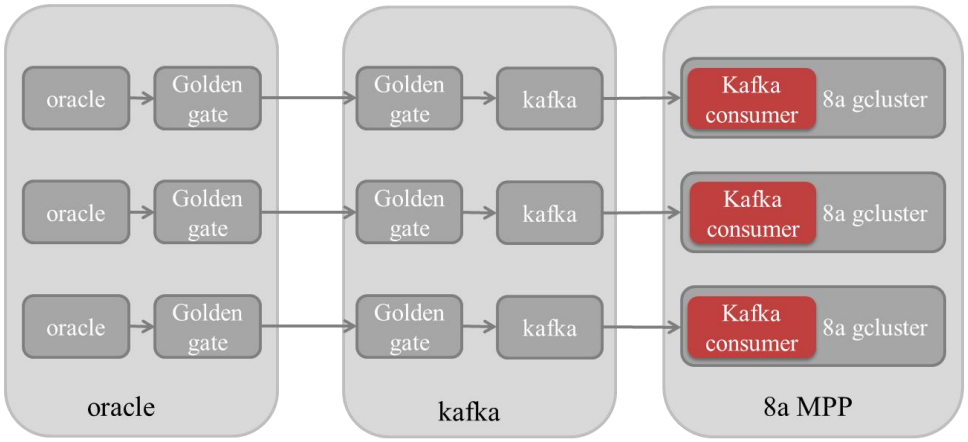
- Highly integrated with the cluster, no additional deployment required;

- Support the loading method of SQL and external tools, and the user oriented SQL interface method unifies cluster loading with data operation methods such as DML;
- Support parallel loading of single table and multiple data sources, and support parallel loading of single table by multiple loading machines to maximize loading performance;
- Support remote reading of data from a universal data server, and support multiple file transfer protocols such as FTP/SFTP/HTTP/HTTPS/HDFS/Kafka;
- Supports loading of data files in various formats such as regular text, gzip compression, snappy compression, and lzo compression;
- Supports normal text mode, fixed length text mode, and loose mode loading;
- Support error data traceability function, which can accurately locate the location of error data in the source data file;
- Support real-time query of loading progress and status;
- The loading performance can continue to improve with the expansion of the cluster size.

2.4.5.2 Kafka data synchronization

The data synchronization system replicates business data from Oracle, GBase 8s, and other databases to the GBase 8a MPP Cluster through tools such as Oracle Goldingate and GBase RTSync. In order to cope with potential peaks in the business system, Kafka message queues are added to the system as buffers. Taking Oracle's synchronization of real-time data to the GBase 8a cluster as an example, the overall process is as follows:

Figure -25 Overall flowchart of Kafka data synchronization



The OGG sender (GoldenGate Extract) extracts transaction information from Oracle's online and archived logs and generates a Trail file. The OGG receiving end (GoldenGate

Replicat) receives the Trail file, extracts transaction information, converts it into the target format, and produces transaction messages to Kafka. The Consumer module of the cluster consumes transaction messages from Kafka and updates the data to GBase 8a MPP Cluster.

The main function of Kafka consumer is to synchronize Kafka data to GBase 8a MPP Cluster:

- According to the configuration, the business that needs to be synchronized ca n be specified;
- Provide synchronization status query function during the synchronization proce ss;
- Achieve high availability and transaction data consistency for data synchroniz ation.

2.4.6 Virtual cluster and mirror cluster

2.4.6.1 Virtual cluster

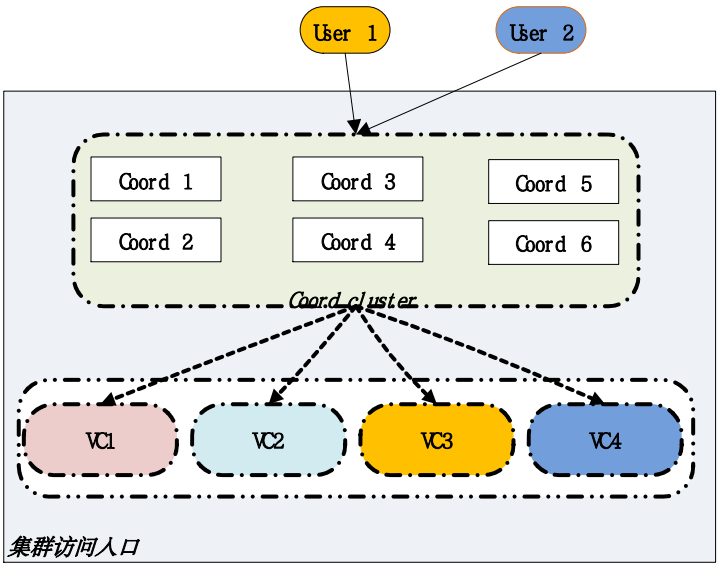
A virtual cluster contains one or more VCs (virtual clusters). Each VC operates independently within the entire virtual cluster, sharing a unified entry point. With permission, each virtual cluster can access each other. Support the deployment of Coordinator nodes and Data nodes on the same physical node.

unified management

Virtual cluster partitioning within the cluster not only enhances the scalability of the cluster, but also provides a unified management view.

Unified entrance

Figure -26 Schematic diagram of cluster access entrance



explain

- The user selects any node in the Coordinator cluster as the cluster entry point;
- The coordinator node connects and operates on the VC specified by the user based on their settings.

Business isolation

Virtual clusters perform vertical resource isolation on clusters, and in actual business scenarios, physical isolation of resources from different businesses can be achieved.

Transparent access

The access of virtual clusters is transparent to applications and is a unified cluster for applications.

2.4.6.2 Mirror cluster

The mirroring function of virtual clusters has the following characteristics:

- Real time: Mirror table data is a real-time backup of data, and any data changes made to either end of the mirror cluster will be synchronized to the mirror table in real time;
- High availability: After the main table has a mirroring relationship, for the query module, it is equivalent to adding backup shards to the main table.

The query module can enhance the high availability of queries by utilizing additional backups;

- Disaster recovery: Support the deployment of mirror clusters in the same city and different locations;

- Easy to maintain: supports creating and deleting image relationships on a library or table basis. When using the library as a unit, all tables under the library will automatically create images; All functions, stored procedures, and views under the library will be synchronized and created under the target image library.

2.4.7 data security

2.4.7.1 data encryption

GBase 8a MPP Cluster data encryption provides soft encryption function for database landing data to meet user security needs and improve system security. The data encryption function provides encryption requirements of different granularity at the table or column level.

Data encryption has the following characteristics:

- Support encrypted keyword encryption for table creation;
- Support encryption requirements of different granularity at the table or column level;
- Support query of table level encryption attributes;
- Support key certificate management: including creation, opening, closing, password modification, and key conversion of key certificates

  Operation;
- Support key type conversion, that is, from plaintext key to ciphertext key, or from ciphertext key to plaintext key

  Key:
  - Clear text key: No user password required, can be randomly generated or manually entered;
  - Cryptographic key: The user must input a password to encrypt and store the randomly generated key based on the password.
- Support querying the current key certificate status;
- Support row and column encryption:
  - Encrypted transmission;
  - Encrypted access;
  - Encrypted storage.

2.4.7.2 Data Desensitization

GBase 8a MPP Cluster provides a new feature of dynamic data desensitization, enabling

developers or database administrator to control the protection method of sensitive data, and generate desensitized data at the database level, greatly simplifying the security design and coding of the database application layer.

By Permission and Field Properties

Users can add desensitization attributes to fields that require data desensitization through SQL syntax, and decide whether to open raw data to users through user permission control.

Built-in rules

Dynamic data desensitization does not truly alter the actual data stored in the table, but only applies this feature to control the data returned during queries. Dynamic data desensitization supports five data desensitization functions, including default desensitization, random desensitization, custom desensitization partial, hash desensitization sha, and specified character position desensitization. Whether dynamic data desensitization is enabled or not is affected by the current user's permissions. Users with unmask permissions are not affected by the desensitization rules and can access actual data. Users without unmask permissions can only access the desensitized data due to the desensitization rules. Desensitization is only effective for projection columns.

2.4.8 Full text search

The GBase 8a MPP Cluster database supports full-text retrieval, adopts a full single word indexing method, supports almost all languages, and can guarantee a 100% query recall rate. Combining the unique column storage, compression, and intelligent indexing technology of GBase 8a MPP Cluster, it is suitable for retrieval and query applications targeting massive data.

The main functions include:

Establishing Indexing and Searching

- Embedded full-text search engine in GBase 8a MPP Cluster, supporting indexing and querying of all text type fields in the table;
- Support parameterized management, and the processes of index establishment, segmentation configuration management, index maintenance, search, etc. can be configured through the standard configuration file of GBase 8a MPP Cluster;

- Embed word segmentation tool function in GBase 8a MPP Cluster to achieve single word segmentation of text columns and search strings, and ensure the consistency of segmentation rules and results between the two, preventing inconsistent segmentation caused by contextual context;
- Supports full-text index synchronization queries, and enables query functionality during the index update process. Newly added data can be indexed in batches. When the data in the index data buffer is processed and written to the index file, users can immediately search for the content of these created indexes, instead of waiting for all new data to be indexed before querying;
- Supports logical expression queries (AND, OR, NOT) and NEAR queries for established full-text indexed columns in database tables, as well as logical combination queries with non full-text indexed fields.

Support DML

- Support the online deletion of established full-text indexes for character data type columns in database tables;
- Support synchronous updates of full-text indexing after column data UPDATE.

Support for DDL

- Support database tables to automatically invalidate the index after the full-text index column is deleted;
- Supports database table renaming without invalidating the index.

2.4.9 Database data mining

GBMLLib is a data mining and machine learning extension library for GBase 8a MPP Cluster, added as a plugin to GBase 8a MPP Cluster. Through its provided machine learning algorithms, GBase 8a MPP Cluster can conduct in-depth analysis and mining of user data, transforming user data into user value.

GBMLLib provides SQL based machine learning algorithms, which currently include regression algorithms (linear regression), classification algorithms (logistic regression, support vector machines), and clustering algorithms (K-Means). It also provides some basic functions for array operations and linear algebra calculation.

GBMLLib has the following technical features:

- SQL interface: GBMLLib provides SQL based data mining algorithms, model training, evaluation, and

  Predictions are executed through SQL statements, making it very easy for data analysts to master. By combining existing skills, they can fully unleash their creativity and improve work efficiency;

- In database analysis: Unlike other analysis tools that require data to be moved from the database to the analysis node through API or ODBC for processing, GBMLLib's analysis algorithm runs in the form of database UDF/UDAF within the threads of GBase 8a MPP Cluster, and is scheduled through the execution plan of GBase 8a MPP Cluster to minimize data movement and improve running speed;

- Convenient expansion: GBMLLib is added to GBase 8a MPP Cluster in the form of a plug-in, and flexible software architecture is used to facilitate the subsequent addition of new data mining and machine learning algorithms.

2.5 Technical indicators

Table -24 Technical indicators

Technical indicators	Description
Cluster can process data	100PB
Single point processable data	100TB
Digital accuracy	65 bit
Number of tables	Dependent file system, such as Ext3 file system, with a maximum of 31998 in a single database; Other file systems are unrestricted.
Number of columns in each table	two thousand
Number of rows in each table	2^{47}
Internal length of each row	65534000 bytes
The length of INTEGER type columns	4 bytes
Number of digits representing the year in the date type column	4 digits
The number of characters in the username	16 characters
CHAR type column length	255 characters
BLOB type column length	32K
Long BlOB type column length	64M
LONG TEXT type column length	64M
VARCHAR type column length	10922 characters UTF8MB4 and GB18030 are 8192 characters, while GBK and UTF-8 are 10922 characters
The length of rows and columns	32KB
Virtual cluster name length	64 characters
Number of virtual clusters	sixty-four
Database name length	48 characters
Table name length	56 characters
Column name length	64 characters
Index name length	64 characters
Alias length	255 characters
Encoding format	UTF-8、UTF8-MB4、GBK 、GB18030

3 Cluster installation, upgrade, and uninstallation

This chapter introduces the specific operation process of installing, upgrading, rolling back, and uninstalling GBase 8a MPP Cluster.

3.1 Preparation of installation environment

3.1.1 Preparing the operating system

Each server has installed the operating system specified in the software requirements section.

The operating system of each node of GBase 8a MPP Cluster needs to meet the following requirements:

Table 31 Operating System Installation Checklist

Inspection items	Inspection content requirements
Operating System Configuration&Software Package	1. Installation mode selection: RedHat 6: Select the "Software Development Workstation" method; RedHat 7: Select "Server with GUI"+"Development Tools"; Centos 8: Select "Server with GUI"+"Development Tools"; SUSE: It is recommended to choose 'c/c++Compiler and Tools'. 2. Can execute the kill all command normally. This command requires support from the psmisc package. The psmisc package is not a default installation package. If it is not installed, it needs to be installed separately to ensure that the kill all command can be executed. 3. Confirm the installation of libcgroup package The libcgroup package is not a default installation package and needs to be installed separately. This package is required by the resource management function. 4. The installed Python version must be Python 2 RedHat 6/7 Python 2 does not need to be installed separately, and is installed by default on the system. Redhat8/Centos8 requires a separate installation of Python 2. After installing Python 2, the command used is Python 2. You need to change the Python 2 command to the default Python command: alternates -- set Python/usr/bin/Python 2 (Centos 8 comes with installation packages for Python 2 and

Inspection items	Inspection content requirements
	Python 3, located in the AppStream directory)
Operating System Version	The operating system versions of nodes within the same VC in the cluster are consistent
Disk partition size and file format	1. Disk partition format: RHEL 6. X: EXT4 file format; RHEL 7. X: XFS file format; SUSE: XFS file format. 2. The disk partition size meets the recommended minimum disk space size requirement or above.
Swap partition settings	1. Size setting: For machines with less than 64GB of memory, it is recommended that Swap and memory be consistent; For machines with more than 64GB of memory, it is recommended to set it to half of the memory or 64GB. 2. Location setting: It is recommended to place Swap files and data files on different disks in the operating system.
CPU configuration	It is recommended to turn off hyper threading and turn off CPU automatic frequency reduction. Opening a hyper thread under high load will increase the waiting time. Meets minimum configuration requirements or above.
Memory	Meets the recommended minimum memory requirements or above.
Host Name Configuration	Meet the requirements of the plan. The host name (domain name) must be less than 46 characters
network	1. The IP configuration of the network card is correct; 2. Network interworking between cluster nodes.
Port number usage	Check that all service default ports in the cluster (see default port list) are not occupied by all nodes in the cluster.
Firewall settings	1. No strong security requirements, close the firewall; 2. There are strong security requirements and permissions to enable the default ports of all services in the cluster (see the default port list and port reference list).
system time	It is required that the system time in the entire cluster be consistent, and it is best to configure clock synchronization.
SSHD Service Status	Start the sshd service of each node normally and confirm the

Inspection items	Inspection content requirements
	port number used by the sshd service.
Virtual memory Configuration	Confirm that the virtual memory configuration mode is unlimited.
Transparent page and I/O scheduling parameter settings	Confirm that the value of the elevator parameter is set to deadline; Transparent_ The hugepate parameter is set to never.
Setting the maximum number of processes allowed by the operating system	For Redhat7. x or SUSE12, defaultTasksMax=infinity needs to be set.
Cluster installation and operation users	1、Confirm that there are installation and running users for each node of the cluster before installation. 2、Confirm that the cluster installation and running users have read and write permissions to the installation directory.
Operating System Environment Check	Execute the configuration script before installation.

The command references for the above checks are as follows, and the default validation version is shown in the table below:

Table -32 Verify Command Operating System Version

operating system	The following reference commands verify the version
RedHat 6	RedHat 6.2
RedHat 7	RedHat 7.3
Centos 8	Centos 8.0
SUSE	SUSE 11 sp3

3.1.1.1 Check operating system version

- RHEL6.X

```
# lsb_ release -a
LSB
Version:    :core-4.0-amd64:core-4.0-noarch:graphics-4.0-amd64:graphics-4.0-n
oarch:printing-4.0-amd64:printing-4.0-noarch
Distributor ID: RedHatEnterpriseServer
Description:    Red Hat Enterprise Linux Server release 6.2 (Santiago)
```

```
Release:        6.2
Codename:       Santiago

#uname -a
Linux gba01 2.6.32-220.el6.x86_ 64 #1 SMP Wed Nov 9 08:03:13 EST 2011 x86_
64 x86
_ 64 x86_ 64 GNU/Linux
```

- RHEL7.X/CENTOS8.X

```
#uname -a
Linux gba01 3.10.0-514.el7.x86_ 64 #1 SMP Wed Oct 19 11:24:13 EDT 2016
x86_ 64 x86_ 64 x86_ 64 GNU/Linux

#cat /etc/redhat-release
Red Hat Enterprise Linux Server release 7.3 (Maipo)
```

- SUSE

```
#uname -a
Linux gba01 3.0.76-0.11-default #1 SMP  Fri Jun 14 08:21:43 UTC
2013(ccab990) x86_ 64 x86_ 64 GNU/Linux

#cat /etc/SuSE-release
SUSE Linux Enterprise Server 11 (x86_64)
VERSION = 11
PATCHLEVEL = 3
```

3.1.1.2 Check disk partition size and format

When installing the system, it is recommended to mount the divided logical volumes to the cluster installation directory (if there are no special requirements in the following use cases,/opt should be used as the product installation directory). The mount configuration of/opt should be written in/etc/fstab instead of/etc/rc.d/rc.local, which can ensure that/opt completes the mount before the 8a cluster starts, otherwise it may cause the 8a cluster to fail to start.

It is recommended to prepare a separate physical disk for the database and do Raid instead of sharing it with the operating system.

Example 1: Checking Disk Partition Size and Partition Format

- RHEL&SUSE (command is universal, the following use case is the execution result of RHEL7 system command)

```
# df -Th
File System Type Capacity Used Free Used% Mount Point
/dev/mapper/rhel-root xfs 17G 3.7G 14G 22% /
devtmpfs devtmpfs 897M 0 897M 0% /dev
tmpfs tmpfs 912M 144K 912M 1% /dev/shm
tmpfs tmpfs 912M 9.0M 903M 1% /run
tmpfs tmpfs 912M 0 912M 0%
/sys/fs/cgroup
/dev/sda1 xfs 1014M 173M 842M 18% /boot
tmpfs tmpfs 183M 8.0K 183M 1% /run/user/0
```

Example 2: Partition Mount: Assuming the partition to be mounted is/dev/sdb1:

- RHEL6. X&SUSE: ext4 file system, adding at the end of the/etc/fstab file

```
/dev/sdb1 /opt ext4
defaults,noatime,nodiratime,nobarrier 0 0
```

- RHEL7. X: xfs file system, adding at the end of the/etc/fstab file

```
/dev/sdb1 /opt xfs
defaults,noatime,nodiratime,nobarrier 0 0
```

3.1.1.3 Check Swap partition

Example: Viewing Swap Partition Size

- RHEL

```
# free -m
              total        used        free      shared  buff/cache
available
Mem:         128652        1893      115499        4234
11258      112560
Swap:        131071          62      131009
```

- SUSE

```
# free -m
              total        used        free      shared  buffers
cached
Mem:          7986        3202        4783         154          0
```

```
2739
-/+ buffers/cache          462          7524
Swap:          1088            0          1088
```

3.1.1.4 Check CPU configuration

Example 1: Viewing the number of CPUs

- RHEL & SUSE:

Total number of CPU cores=number of physical CPU cores x number of physical CPUs

Total number of logical CPUs=total number of cores x number of hyper threads

```
--Viewing the number of physical CPUs
#cat /proc/cpuinfo| grep "physical id"| sort| uniq| wc -l
or
#grep 'physical id' /proc/cpuinfo |sort -u|wc -l
--View the number of logical CPUs
#cat /proc/cpuinfo| grep processor|sort|uniq|wc -l
or
#grep -c processor /proc/cpuinfo
--View the number of siblings, the number of logical CPUs contained in a
physical CPU
#grep "siblings" /proc/cpuinfo|uniq
--View the number of CPU cores, the number of cores in a physical CPU
#grep "cpu cores" /proc/cpuinfo|uniq
--View the number of core IDs
# grep 'core id' /proc/cpuinfo
```

Based on the above query results, if "siblings" and "CPU cores" are consistent, it indicates that hyperthreading is not supported or not opened; If 'siblings' is twice the size of' CPU cores', it indicates that hyper threading is supported and enabled. If two logical CPUs have the same 'core id', then the hyper thread is open.

Example 2: Viewing CPU Models

- RHEL & SUSE:

```
# grep 'model name' /proc/cpuinfo
model name      : Intel(R) Core(TM)2 Duo CPU     E7500  @ 2.93GHz
```

```
model name      : Intel(R) Core(TM)2 Duo CPU     E7500   @ 2.93GHz
```

3.1.1.5 Check memory condition

- RHEL & SUSE:

```
# grep 'MemTotal' /proc/meminfo
MemTotal:       4050180 kB
```

3.1.1.6 Configure Host Name

Example: Assuming the current IP is 172.168.83.11 and the hostname to be changed is gba01, the specific setting method is as follows:

- RHEL6.X

1. Modify the/etc/hosts file and add relevant information;

```
# vi /etc/hosts
127.0.0.1 localhost localhost. localdomain localhost4 localhost4.
localdomain4//Reserved
: 1 localhost localhost. localdomain localhost6. localdomain6//Reserved
172.168.83.11 www.gba01.com gba01
(: wq save exit)
```

2. Modify the content of the/etc/sysconfig/network file;

```
# vi /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=gba01
(: wq save exit)
```

3. Restart

```
#shutdown -r now
```

- RHEL7.X、Centos8.X

```
#hostnamectl set-hostname gba01
```

- SUSE

1、 Modify the content of the/etc/HOSTNAME file;

```
# vi /etc/HOSTNAME
gba01
(: wq save exit)
```

2、 restart

```
#reboot
```

3.1.1.7 Network configuration requirements: Confirm whether the IP settings are correct

Example 1: IP Setting Check

- RHEL && SUSE

```
# ifconfig
eth0        Link encap:Ethernet  HWaddr 00:0C:29:2B:93:C1
            inet addr:172.168.83.11  Bcast:192.168.255.255
Mask:255.255.0.0
            inet6 addr: fe80::20c:29ff:fe2b:93c1/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500
Metric:1
            RX packets:38240 errors:0 dropped:0 overruns:0 frame:0
            TX packets:341 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:3568806 (3.4 MiB)  TX bytes:41599 (40.6 KiB)
```

- RHEL 7.X、Centos8.X

```
# nmcli dev
```

Example 2: Network IP settings focus on the settings in the following example.

- RHEL 6.X

```
# vi /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
HWADDR=00:0C:29:2B:93:C1
NM_ CONTROLLED=yes
ONBOOT=yes
IPADDR=172.168.83.11
BOOTPROTO=none
NETMASK=255.255.0.0
TYPE=Ethernet
IPV6INIT=no
USERCTL=no
```

- RHEL 7.X:

```
# vi /etc/sysconfig/network-scripts/ifcfg-ens33
TYPE=Ethernet
BOOTPROTO=none
NAME=ens33
DEVICE=ens33
ONBOOT=yes
IPADDR=172.168.83.11
NETMASK=255.255.0.0
GATEWAY=172.168.1.0
DNS1=172.168.1.255
```

- CentOS 8.X

```
# vi /etc/sysconfig/network-scripts/ifcfg-ens33
TYPE=Ethernet
BOOTPROTO=static
IPADDR=192.168.146.150
NETMASK=255.255.255.0
PREFIX=24
GATEWAY=192.168.146.254
NAME=eth33
UUID=8667e896-719c-42df-8635-f9d36a4a4c17
DEVICE=enp0s17
ONBOOT=yes
MACADDR=00:0C:29:CF:19:6A
```

- SUSE

```
# vi /etc/sysconfig/network/ifcfg-eth0
BOOTPROT0=’static’
IPADDR=’172.168.83.11’
NETMASK=’255.255.0.0’
NETWORK=’172.168.1.0’
BROADCAST=’172.168.1.255’
```

3.1.1.8 Check node port number

Check if the default ports used by each node and service are occupied.

The default ports used by GBase 8a MPP Cluster services are as follows:

Table 33: Description of default ports used by various services in GBase 8a MPP Cluster

Component Name	Default Port Number	Port protocol type	Port Meaning
Gcluster	five thousand two hundred and fifty-eight	TCP	The port on which GCluster cluster nodes provide external services
Gnode	five thousand and fifty	TCP	The port on which the data cluster node provides external services
Gcware	five thousand nine hundred and eighteen	TCP/UDP	Communication port between gcware nodes
gcware	five thousand nine hundred and nineteen	TCP	External connection gcware node port
syncServer	five thousand two hundred and eighty-eight	TCP	SyncServer service port
GcrecoverMonit	six thousand two hundred and sixty-eight	TCP	Gcrecover service port
Remote data export port	16066-16166	TCP	Remote data export port

 **be careful**

1、The port requirements for all Gcluster cluster nodes are consistent

2、The port requirements for all data cluster nodes are consistent

3、 The port requirements for all gcware cluster nodes are consistent

Example 1: Check whether the ports used by each service on each node are occupied (command "lsof - i: PORT").

- RHEL && SUSE

```
# lsof -i:5258
COMMAND     PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
gclusterd 1791 gbase   14u  IPv4  12511      0t0  TCP *:5258 (LISTEN)
```

3.1.1.9 Firewall settings

Example 1: Checking the firewall status

- RHEL6. X: Execute the following command to view the firewall status of each node:

```
# service iptables status
iptables: Firewall is not running.

# service ip6tables status
ip6tables: Firewall is not running.

//The following information is displayed to indicate that the firewall is turned on:
# service iptables status
Table: nat
Chain PREROUTING (policy ACCEPT)
num  target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
num  target     prot opt source               destination
1    MASQUERADE  tcp  --  192.168.122.0/24     ! 192.168.122.0/24
masq ports: 1024-65535
2    MASQUERADE  udp  --  192.168.122.0/24     ! 192.168.122.0/24
masq ports: 1024-65535
3    MASQUERADE  all  --  192.168.122.0/24     ! 192.168.122.0/24

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination

Table: mangle
Chain PREROUTING (policy ACCEPT)
num  target     prot opt source               destination

Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
```

```
Chain FORWARD (policy ACCEPT)
num   target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
num   target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
num   target     prot opt source               destination
1     CHECKSUM   udp  --  0.0.0.0/0            0.0.0.0/0
udp dpt:68 CHECKSUM fill

Table: filter
Chain INPUT (policy ACCEPT)
num   target     prot opt source               destination
1     ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp
dpt:53
2     ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp
dpt:53
3     ACCEPT     udp  --  0.0.0.0/0            0.0.0.0/0           udp
dpt:67
4     ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp
dpt:67

Chain FORWARD (policy ACCEPT)
num   target     prot opt source               destination
1     ACCEPT     all  --  0.0.0.0/0            192.168.122.0/24    state
RELATED,ESTABLISHED
2     ACCEPT     all  --  192.168.122.0/24     0.0.0.0/0
3     ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
4     REJECT     all  --  0.0.0.0/0            0.0.0.0/0
reject-with icmp-port-unreachable
5     REJECT     all  --  0.0.0.0/0            0.0.0.0/0
reject-with icmp-port-unreachable

Chain OUTPUT (policy ACCEPT)
```

```
num   target     prot opt source               destination

# service ip6tables status
Table: filter
Chain INPUT (policy ACCEPT)
num   target     prot opt source               destination
1     ACCEPT     all      ::/0                 ::/0
state RELATED,ESTABLISHED
2     ACCEPT     icmpv6   ::/0                 ::/0
3     ACCEPT     all      ::/0                 ::/0
4     ACCEPT     tcp      ::/0                 ::/0
state NEW tcp dpt:22
5     REJECT     all      ::/0                 ::/0
reject-with icmp6-adm-prohibited

Chain FORWARD (policy ACCEPT)
num   target     prot opt source               destination
1     REJECT     all      ::/0                 ::/0
reject-with icmp6-adm-prohibited

Chain OUTPUT (policy ACCEPT)
num   target     prot opt source               destination
```

Execute the following command to check if the firewall automatically starts when turned on:

```
# chkconfig --list iptables
Iptables 0: Off 1: Off 2: Off 3: Off 4: Off 5: Off 6: Off
# chkconfig --list ip6tables
Ip6tables 0: Off 1: Off 2: Off 3: Off 4: Off 5: Off 6: Off
```

- RHEL7. X, Centos8. X: Execute the following command to view the firewall status of each node:

```
#systemctl status firewalld.service
```

- SUSE: Execute the following command to view the firewall status of each node:

```
#rcSuSEfirewall2 status
```

Example 2: Closing the firewall

- RHEL6. X: Execute the following command to close the firewall. The specific command is as follows:

 # service iptables stop

iptables: Flushing firewall rules: [OK]

iptables: Setting chains to policy ACCEPT: filter [OK]

iptables: Unloading modules: [OK]

 # chkconfig iptables off

Description: If this command execution is invalid, the following command can be executed:

 # chkconfig iptables off --level 2345

 # service ip6tables stop

ip6tables: Flushing firewall rules: [OK]

ip6tables: Setting chains to policy ACCEPT: filter [OK]

ip6tables: Unloading modules: [OK]

 # chkconfig ip6tables off

Description: If this command execution is invalid, the following command can be executed:

 # chkconfig ip6tables off --level 2345

- RHEL7.X、Centos8.X

--Turn off firewall

#systemctl stop firewalld.service

--Prohibit firewall startup

#systemctl disable firewalld.service

- SUSE 11

--The closing operation is:

#service SuSEfirewall2_ setup stop

#service SuSEfirewall2_ init stop

--Cancel startup of firewall:

#chkconfig SuSEfirewall2_ init off

#chkconfig SuSEfirewall2_ setup off

- SUSE 12

--The closing operation is:

#systemctl stop SuSEfirewall2.service

--Cancel startup of firewall:

```
#systemctl disable SuSEfirewall2.service
```

Example 3: Open Port Number:

The command reference for setting port rules and developing ports is as follows:

- RHEL6.X

1. Set default rules.

```
# iptables -A INPUT -j DROP
# iptables -A FORWARD -j ACCEPT
```

2. Open port, PORT is the port number to be opened.

```
--Open TCP port
# iptables -I INPUT -p tcp --dport PORT -j ACCEPT
# iptables -I OUTPUT -p tcp --dport PORT -j ACCEPT
# iptables -I INPUT -p tcp --sport PORT -j ACCEPT
# iptables -I OUTPUT -p tcp --sport PORT -j ACCEPT

--Open UDP port
# iptables -I INPUT -p udp --dport PORT -j ACCEPT
# iptables -I OUTPUT -p udp --dport PORT -j ACCEPT
#iptables -I INPUT -p udp --sport PORT -j ACCEPT
#iptables -I INPUT -p udp --sport PORT -j ACCEPT
```

- RHEL7.X、Centos8.X

```
--View ports that have already been opened
#firewall-cmd --list-ports
--Open port (after opening, the firewall needs to be restarted to take effect), and
PORT is the port number to be opened
#firewall-cmd --zone=public --add-port=PORT/tcp --permanent
--Restarting the firewall
#firewall-cmd --reload
```

- SUSE

1. Manually modify the/etc/sysconfig/SuSEfirewall2 configuration file (PORT1 and PORT2 represent multiple ports to be opened):

```
#vi /etc/sysconfig/SuSEfirewall2
#TCP Port
FW_ SERVICES_ EXT_ TCP = "PORT1 PORT2"
```

```
#UDP port
FW_ SERVICES_ EXT_ UDP = "PORT1 PORT2"
```

2. service iptables restart

```
#rcSuSEfirewall2 restart
```

3.1.1.10 Check the sshd service status

Example 1: Checking the service startup status

- RHEL6.X && SUSE

```
--Set the sshd service to automatically load and run after each startup
# chkconfig sshd on
--Check if the sshd service is enabled
# chkconfig --list sshd
```

- RHEL7.X、Centos8.X

```
--Set the sshd service to automatically load and run after each startup
#systemctl enable sshd.service
--Check if the sshd service is enabled
# systemctl status sshd.service
```

3.1.1.11 Check the virtual memory configuration

- RHEL && SUSE

Method 1: Permanently take effect: Use the ROOT user to modify the configuration file/etc/security/limits.conf, add the following two lines, and it will take effect after the operating system restarts:

```
#vi /etc/security/limits.conf
*    soft as unlimited
*    hard as unlimited
# reboot
```

Method 2: Current system takes effect: Execute the following command to modify the current system configuration. After restarting the operating system, it needs to be reset:

```
# ulimit -H -v unlimited

# ulimit -S -v unlimited
```

Table -34 Parameter Description

Parameter Name	Description
-H	Set hard limits on resources, which are set by administrators.
-S	Set elastic limits on resources.
-V<virtual memory size>	Specifies the upper limit of available virtual memory in KB.

3.1.1.12 Check transparent large pages and adjust I/O scheduling parameters

- RHEL6.X

Method 1: Command line modification

- Modify file/etc/default/grub
- Find GRUB_ CMDLINE_ On the LINUX line, add elevator=deadline transparent within double quotes_ hugepage=never
- Save Exit
- Then use the operating system root user to execute grub2-mkconfig -o/boot/grub2/grub.cfg
- Reboot

Method 2. The script modification content can be as follows

```
#####Modify the transparent page transparent as follows_ Hugepage parameters and disk IO scheduling################
cp /etc/default/grub /etc/default/grub_ bak
line_ num=`cat -n /etc/default/grub|grep 'GRUB_ CMDLINE_ LINUX'|awk '{print $1}'|head -n 1`
sed -i --follow-symlinks 's/elevator=deadline//g' /etc/default/grub
sed -i --follow-symlinks 's/transparent_ hugepage=never//g' /etc/default/grub
sed -i --follow-symlinks ""${line_num}"s/\"$/ elevator=deadline\"/g" /etc/default/grub
Echo "Parameter elevator=deadline modified"
sed -i --follow-symlinks ""${line_num}"s/\"$/ transparent_
```

```
hugepage=never\"/g" /etc/default/grub
Echo "The parameter transparent hugepage has been modified"
#########Execute the following command to make it effective#########################
grub2-mkconfig -o /boot/grub2/grub.cfg
###The following is a restart of this machine
reboot
```

be careful

Executing this script will automatically restart the local machine

3.1.1.13 Setting the maximum number of processes allowed by the operating system

- RHEL 7 & SUSE 12

```
# vi /etc/systemd/system.conf
DefaultTasksMax=infinity
(: wq exit)
#reboot
```

3.1.2 Prepare the GBase 8a MPP Cluster software installation package

Obtain the corresponding GBase 8a MPP Cluster software installation package based on the version of the operating system (the same cluster uses the same version of the operating system).

The software installation package version rules are as follows:

```
GBase8a_ MPP_ Cluster-[No]License-9.5.3.xx-OSversion-platform.tar.bz2
```

Table -35 GBase 8a MPP Cluster Software Installation Package

Software package files	operating system	Acquisition method
GBase8a_ MPP_ Cluster-[No]License-9.5.3.X-SUSExx-x86_ 64.tar.bz2	Suse	Apply for software installation packages and manuals through sales or pre-sales personnel at Nanjing University General Motors.
GBase8a_ MPP_ Cluster-[No]License-9.5.3.X-redhatxx-x86_ 64.tar.bz2	Redhat	Apply for software installation packages and manuals through sales or pre-sales personnel at Nanjing University General Motors.

3.2 install

explain

Unless otherwise specified, the installation cases in this chapter shall follow the following rules:

- The cluster installation user is' gbase ';
- The version of the cluster installation case is "GBase8a MPP_Cluster NoLicense-9.5.3. * - redhat7.3-x86_64";
- The cluster installation directory is'/opt ';
- Multiple instance installation requires each network card of the server with multiple network cards to have an independent IP, and each IP has one instance. The installation steps are the same as for a single instance;
- Related to database installation:
  - After installation, a/home/gbase/. gbase will be generated_ Profile file is used to set environment variables
  - After installing the GBase 8a MPP Cluster database, there will be two default database users, root and gbase. The root user password is initially blank and is the superuser of the database. GBase is a database user commonly used for database management operations such as backing up gcrcman, gcadmin, etc. to connect to the server. For the first time logging into the database, please use the root user. Under the root user, change the root's own password in a timely manner and set the password for the gbase user. Keep the root and gbase passwords properly, so that subsequent database management operations can connect to the server normally.

    Note: The database password in the manual is represented by *******.
  - After installing the GBase 8a MPP Cluster database, the relevant directories and functional descriptions are as follows. Each cluster node creates its own IP named subfolder $GCLUSTER in the installation directory_ PREFIX (such as opt/IP), the main directories inside are as follows:

Table -36 Relevant directories and functions involved in the operation of database software services

Directory Name	Directory Function Description
gcware	This directory is located in the installation directory and is used to store related files of gcware, including configuration files and logs. The gcware node has this directory.
gcluster	This directory is located in the installation directory and is used to store related files of gcluster, including configuration files, logs, and metadata.
gnode	This directory is located in the installation directory and is used to store relevant files of gnode, including configuration files, logs, and user data.
/home/dbaUser/	.gbase_ The profile file is an environment variable file for the database installation user.

3.2.1 Obtain License

The license file is a required parameter during cluster installation, so it must be obtained before cluster installation.

The cluster license certification process is as follows:

1. Obtain ESN information. ESN is not related to cluster server hardware, it identifies a cluster and is globally unique. The length is between 40-580 characters.
2. Send ESN information and license restriction requirements information to the original factory of Nanjing University General Motors
3. Obtain the license file from the original factory response
4. Use this license file for cluster installation
5. Set the periodic inspection time as needed

The license is bound to the gcware node and depends on the private/public key of the gcware node. License is the gcware node installed in the cluster, which generates an ESN and stores the license, saving the list of valid nodes in the cluster. So in the following situations, a new license needs to be applied for:

1) There is a change in the private/public key of the gcware node. ESN is generated based on the user's SSH private key, and attention should be paid to protecting and backing up the private key of the gcware node running the user.
2) After replacing the Gcware node, a new license needs to be applied for.
3) When upgrading, a new license needs to be applied for.

License type description:

The default license does not require an ESN when applying, and can be generated directly according to restricted requirements.

Both trial and commercial licenses need to provide an ESN, generated according to ESN and restriction requirements.

License restrictions require support for the following:

Expiration date, license days, scale license (number of gnodes), capacity license (total data capacity of all gnode nodes).

Note: After the cluster reaches the capacity limit value, the cluster state will be set to shrinkOnly, which restricts insert and load operations. Users can reduce space through operations such as drop table, and then execute the show license command. This command will refresh the capacity and update the cluster state.

3.2.1.1 Generate SSH private key

Operating Steps

step

Users running gcware on Gcware nodes generate SSH private keys in advance and save them in the default location. The generation of ESN requires obtaining the public key from the SSH private key for calculation. When verifying legitimacy, use the calculated ESN node fingerprint.

SSH private key generation using Linux's openssh built-in tool:

Execute the following command using gbase on all gcware nodes as a user

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key(/root/.ssh/id_rsa):
```

Pay attention to protecting and backing up the SSH private key. If the private key is changed, it will cause the license file for this batch of applications to become invalid.

3.2.1.2 Obtain ESN

Operating Steps

Enter the cluster installation package decompression directory gcinstall and execute the following command:

```
$ cd gcinstall
$ ./ getesn.py –silent=<demo.options>
```

be careful

When there are multiple gcware nodes, generating ESNs multiple times is independent of the node configuration order. ESNs are only related to the node's private key/public key, and as long as the private key/public key remains unchanged, the generated ESNs remain unchanged. The log of the acquisition process can be viewed in the getesn.log file in the same directory as the program getesn.

3.2.1.3 Apply for License

Operating Steps

step

provide for license@gbase.cn Provide information and apply, including project number, expiration date, license days, license scale, capacity license, license category, ESN, and

product name.

Before installation, you must obtain a license, which is a required parameter during installation.

Table -312 Parameter Description

Parameter Name	Description
Project number	2-20 digits long.
License Type	Default license, trial license, commercial license The default license does not require submitting an ESN
Product Name	The product name of GBase 8a, including cluster, standalone, and version number
License Days	The value must be greater than 0, unlimited means unrestricted
Expiration date	YYYY-MM-DD/unlimited Do not include time zone information, only consider dates. Unlimited means unrestricted
Scale License	The number of nodes in gnode, with a value greater than 0 as an integer. Unlimited indicates no limit
Capacity licensing	Unit GB/TB/PB can be written. The capacity is the used capacity of the file system where all data nodes' table spaces are located, with unlimited indicating unlimited capacity
ESN	The unique identifier of the cluster needs to be generated in advance on the server where the cluster needs to be installed using the tools provided in the cluster installation package.

After obtaining the license file from the original factory, you can execute commands such as cluster installation and upgrade.

For example, cluster installation commands:

```
$ gcinstall.py –license_ file=LICENSE_ FILE –silent=demo.options
```

explain

The installation process will traverse the gcware nodes for license verification to ensure that the license is legal. After verification, the license file will be saved in the home directory of the gcware operating user.

3.2.1.4 Check LICENSE

Operating Steps

Steps

Users can obtain license file information and ESN information by using the following

command:

checkLicense
perhaps
show license
When executing the show license, the capacity will also be refreshed and checked.

The displayed license information includes: license type, product name, ESN, issuing company, issuing time, license days, expiration date, cluster size limit, and cluster capacity limit. If there is no license, display None.

3.2.1.5 Set cycle inspection time

The cluster will periodically check whether the license restrictions are met. If no check cycle is set, the default check time is 21:00 every day. You can set the inspection cycle as needed by using the following command:

gcadmin setcolltime HH:MM
You can use the following command to view the cycle check time:
gcadmin showcolltime

3.2.1.6 Application after License Expiration or Expiration

The application steps after the license expires or expires are the same as the new cluster application process and method. After applying for a new license file, execute the following command to update the license file:

importLicense *FILENAME*

The cluster uses gcware's operating user to execute, and the license file needs to have access permissions for the executing user. The execution process will recalculate the node fingerprint and compare it with the fingerprint in the license for legitimacy verification.

3.2.2 Initial installation

Operating Steps

Please obtain the license file before performing the following installation steps.

Step 1: Obtain the installation package and extract it:

1. Copy the installation package to a directory on the file system, and the copy command reference is as follows:

```
# cp /tmp/GBase8a_ MPP_ Cluster-NoLicense-9.5.3.17-redhat7.3-x86_
64.tar.bz2 /opt
```

2. Enter the directory and use the tar command in command line mode to decompress. The decompression command is as follows:

```
#cd /opt
# tar xjf  GBase8a_ MPP_ Cluster-NoLicense-9.5.3.17-redhat7.3-x86_64.tar.bz2
```

3. After decompression, the gcinstall directory will be generated in the decompression directory:

```
# ls /opt
gcinstall
```

Step 2: Create a DBA user and configure permissions

1. Create DBA users on all cluster node servers using the operating system root user.

In the installation example, GBase is used as an example for DBA users. Unless otherwise specified in this manual, GBase is used as the default DBA user.

```
# useradd gbase
# passwd gbase
```

2. Change the owner of the gcinstall directory to a DBA user using the root user

```
# chown –R gbase:gbase gcinstall
```

3. Use root user to change the owner of the installation directory to DBA user

The installation directory is the software installation directory specified by the installPrefix parameter in the demo.options file, which defaults to/opt

```
# chown –R gbase:gbase /opt
```

4. Use root user to grant DBA users permission to install GBase on all nodes

Use root to copy the SetSysEnv.py file in the gcinstall directory to all node servers in the cluster and execute the file.

```
# scp SetSysEnv.py  root@192.168.146.21 :/opt
# /opt/SetSysEnv.py  --dbaUser=gbase --installPrefix=/opt
```

SetSysEnv.py syntax description:

```
# python SetSysEnv.py --dbaUser=* --installPrefix=* [--cgroup]
```

Parameter Name	Description
--installPrefix=INSTALLPREFIX	The user can configure the installation directory, which must be installPrefix in demo. options. Cluster logs use this parameter according to the archive function.
--dbaUser=DBAUSER	Must be a dbaUser in demo. options.
--cgroup	When using the resource management function, it is mainly used to modify the resource management configuration file. Must be executed before using resource management.

 **be careful**

Before installation, it is necessary to use root to execute the one click deployment script SetSysEnv.py provided in the installation package on the gcluster node and gnode node. If the GCware node is deployed independently on a separate server, the GCware node does not need to execute the SetSysEnv.py file.

1. Copy the script to each node of the cluster to be installed, and each node needs to be executed using root;
2. Before installing each node in the cluster, there must be an installation user for the cluster and have read and write permissions to the installation directory

Step 3: Switch to DBA user and modify the installation configuration file parameters.

Enter the extracted gcinstall directory and modify the installation parameter file demo.options based on the actual cluster environment. The specific steps are as follows:

```
#su - gbase
$ vi /opt/gcinstall/demo.options
installPrefix= /opt
coordinateHost = 192.168.146.20,192.168.146.21,192.168.146.22
coordinateHostNodeID = 20,21,22
dataHost =
192.168.146.20,192.168.146.21,192.168.146.22,192.168.146.23,192.168.146.40,1
92.168.146.41,192.168.146.42,192.168.146.43
#existCoordinateHost =
#existDataHost =
#existGcwareHost=
gcwareHost = 192.168.146.20,192.168.146.21,192.168.146.22
gcwareHostNodeID = 20,21,22
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbase'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
#characterSet = utf8
#dbPort = 5258
#sshPort = 22
```

Table -37 Parameter Description

Parameter Name	Description

Parameter Name	Description
installPrefix	Specify the installation directory;
coordinateHost	A list of all cluster scheduling nodes (GCluster nodes), which can be IPV4, IPV6, host name, or domain name. However, it is not recommended to deploy the cluster using a mixed method of IPV4 and IPV6, with addresses separated by ',';
coordinateHostNodeID	The nodeid of each GCluster node must be specified during IPV6 and domain name installation, but may not be specified during IPV4 installation. It corresponds one-to-one to the node list listed in coordinateHost, separated by ",";
dataHost	The list of all cluster data nodes can be IPV4, IPV6, host name, or domain name, but it is not recommended to deploy the cluster using a mixed method of IPV4 and IPV6, with node addresses separated by ','; When deploying a cluster using the domain name method, it is necessary to first configure the correspondence between the domain name and IP in the/etc/hosts of each node.
existCoordinateHost	A list of all existing coordinator nodes, separated by ',' between IP addresses, used when installing new nodes for cluster expansion;
existDataHost	A list of all existing Data node IPs, separated by ',' between IP addresses, used when installing new nodes for cluster expansion;
existGcwareHost	A list of all existing Gcware node IPs, separated by "," between IP addresses, used when installing new nodes for cluster expansion;
gcwareHost	All cluster gcware nodes can be IPV4, IPV6, host name, domain name, and must be specified. Separate addresses with ',';
gcwareHostNodeID	The nodeid of each gcware node must be specified during IPV6 and domain name installation, but not during IPV4 installation. It corresponds one-to-one to the node list listed in gcwareHost, separated by ",";
dbaUser	The administrator user of the cluster, the operating system username used when installing and managing the cluster. Before installing, expanding, or replacing nodes, all servers in the cluster must first create a user with the same username as the cluster administrator user;
dbaGroup	The group name to which the operating system user belongs during cluster installation and runtime;
dbaPwd	The password of the operating system user used during cluster

Parameter Name	Description
	installation and runtime;
rootPwd	Install the unified password for the root user of the Linux operating system on the cluster node machine. The cluster still requires root privileges in versions that do not support non root user installations, and when upgrading these versions, the user root password is still required. Other situations do not require use;
rootPwdFile	This parameter supports root users with different password methods on multiple nodes, and cannot be used simultaneously with the parameter rootPwd, otherwise an error will be reported. Cluster versions that do not support installation by non root users still require root privileges, so when upgrading these versions, the user root password is still required. Other situations do not require use;
characterSet	The system supports the installation of specified character sets, which are not displayed in the demo. options file by default. The default character set for 953 is utf8mb4 (4 bytes), with optional values of [utf8, gbk, gb18030, utf8mb4].
sshPort	Used to specify the sshd service port numbers of servers on each node of the cluster, and it is required that all nodes have the same sshd service port numbers. The default value is 22.

Users can modify the above parameters according to actual situations.

be careful

- The above parameters related to passwords, if the setting value contains special symbols, need to use a single quotation mark "'" as the enclosing character, without the need for escape;
- The mixed mode of IPV4 and IPV6 is not supported;

  The writing mode of IPV6 must be consistent, either abbreviated or fully written, and does not support mixing of abbreviated and fully written;

  The writing mode of ipv6 during cluster upgrade, expansion, scaling, uninstallation, and node replacement must be consistent with the writing mode of ipv6 during installation.
- Parameter dbaUser, dbaGroup usage specification:

  1. Supports single quotation marks and double quotation marks as delimiters;
  2. The parameter values follow the naming rules of the Linux system.
- The host name (domain name) must be less than 46 characters

Step 4: GBase DBA user performs installation:

Enter the installation directory and use the DBA user to execute the installation script gcinstall.py.

be careful

- Gcinstall.py must be executed on any coordinator node for installation;
- If it is a license version, you must obtain the license license file in advance, such as

  GBase8a_ MPP_ Cluster-License-9.5.3.27.*-redhat7-x86_ 64.tar.bz2

The specific command syntax is as follows:

./gcinstall.py --license_ file=*licenseFile* --silent=*demo.options*

[--passwordInputMode]

Table 38 Parameter Description

Parameter Name	Description
--silent	Specify the configuration file for cluster installation;
--license_ file	Original factory license file, write the full path and file name. The file needs to be obtained before installation and has access permissions for the user executing the cluster installation.
--passwordInputMode	The gcinstall.py command is an optional parameter. Used to specify the method of password acquisition, and achieve different acquisition methods through different parameters. If this parameter is specified, the password in demo. options does not need to be modified again. The value range is file, pwdsame, pwddiff, and the default value is file. The value description is as follows: ● File: represents obtaining from a file or command line parameter, consistent with the original method, in which the password in the file is plaintext; ● Pwdsame: This parameter is used when the password is entered by the user from the terminal and the passwords of all nodes are consistent. For different user passwords, only one input is required; ● Pwddiff: This parameter is used when the password is entered by the user from the terminal and the passwords between nodes are inconsistent. For different user passwords, each node is input once.

Example:

The example version is GBase8a_ MPP_ Cluster-NoLicense-9.5.3.*-redhat7.3-x86_ 64. tar. bz2, so the command was executed without a license_ The file parameter, if the actual version obtained is the license version, such as GBase8a_ MPP_

Cluster-License-9.5.3.27.*-redhat7-x86_ 64. tar. bz2, then license_ File is a required parameter.

```
$ ./ gcinstall.py    --silent=demo.options
**********************************************************************
Thank you for choosing GBase product!
……
192.168.146.40 install cluster on host 192.168.146.40 successfully.
192.168.146.41 install cluster on host 192.168.146.41 successfully.
192.168.146.42 install cluster on host 192.168.146.42 successfully.
192.168.146.43 install cluster on host 192.168.146.43 successfully.
192.168.146.22 install gcware and cluster on host 192.168.146.22 successfully.
192.168.146.23 install cluster on host 192.168.146.23 successfully.
192.168.146.20 install gcware and cluster on host 192.168.146.20 successfully.
192.168.146.21 install gcware and cluster on host 192.168.146.21 successfully.
Starting all gcluster nodes ...
adding new datanodes to gcware ...
##The above message indicates successful installation
```

Step 5: Post installation status:

After successful installation, you can view the cluster status through gcadmin.

Before executing the gcadmin command, you need to first execute source~/. bash_ Profile makes environment variables effective.

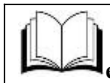
- After installation, the cluster status can be viewed, and all services are open.

```
$ source ~/.bash_ profile
$ gcadmin
CLUSTER STATE:              ACTIVE


=======================================
|   GBASE GCWARE CLUSTER INFORMATION    |
=======================================
| NodeName |      IpAddress       | gcware |
---------------------------------------
| gcware1    | 192.168.146.20 |    OPEN    |
---------------------------------------
| gcware2    | 192.168.146.21 |    OPEN    |
---------------------------------------
```

```
| gcware3    | 192.168.146.22 |    OPEN    |
---------------------------------------
=============================================================
|              GBASE COORDINATOR CLUSTER INFORMATION              |
=============================================================
|     NodeName     |      IpAddress       | gcluster | DataState |
-----------------------------------------------------------
| coordinator1 | 192.168.146.20 |     OPEN     |         0         |
-----------------------------------------------------------
| coordinator2 | 192.168.146.21 |     OPEN     |         0         |
-----------------------------------------------------------
| coordinator3 | 192.168.146.22 |     OPEN     |         0         |
-----------------------------------------------------------
=============================================================
====
|                GBASE CLUSTER FREE DATA NODE INFORMATION
|
=============================================================
====
| NodeName    |     IpAddress       | gnode | syncserver | DataState |
-------------------------------------------------------------
| FreeNode1 | 192.168.146.21 | OPEN    |      OPEN       |         0         |
-------------------------------------------------------------
| FreeNode2 | 192.168.146.41 | OPEN    |      OPEN       |         0         |
-------------------------------------------------------------
| FreeNode3 | 192.168.146.20 | OPEN    |      OPEN       |         0         |
-------------------------------------------------------------
| FreeNode4 | 192.168.146.40 | OPEN    |      OPEN       |         0         |
-------------------------------------------------------------
| FreeNode5 | 192.168.146.23 | OPEN    |      OPEN       |         0         |
-------------------------------------------------------------
| FreeNode6 | 192.168.146.43 | OPEN    |      OPEN       |         0         |
-------------------------------------------------------------
| FreeNode7 | 192.168.146.22 | OPEN    |      OPEN       |         0         |
-------------------------------------------------------------
```

```
| FreeNode8 | 192.168.146.42 | OPEN   |      OPEN      |       0       |
------------------------------------------------------------


0 virtual cluster
3 coordinator node
8 free data node
```

3.2.3 Initial configuration



explain

- When using a single VC, it is recommended to use compatibility mode. Refer to the "3.2.2.1 Initial Configuration of Single VC Mode (Compatibility Mode)" section for configuration
- For multi VC, refer to "3.2.2.2 Initial Configuration of Multi VC Mode"
- Single VC mode is a mode that bypasses the manual creation of VC steps and directly creates a distribution. If you manually create and add VC in single VC mode, the cluster will automatically switch to multi VC mode; The multi VC mode does not support fallback to single VC mode. If the newly added VC is deleted and only the original VC is retained, the cluster mode will still be multi VC mode.
- All operating system users in this chapter use DBA users

3.2.3.1 Single VC mode (compatibility mode) initial configuration

Related concepts

V95 supports the installation and configuration method of V86. The V95 cluster installed using the installation and configuration method of V86 belongs to the single VC mode of V95, also known as "compatibility mode", which means it is compatible with the installation configuration and usage habits of the V86 version. That is, the installation process does not require the user to manually create a VC, and the cluster automatically generates a default VC with vcname of vcname00000 1. The cluster defaults to all current nodes belonging to that VC.

Upgrading from versions prior to GBase 8a MPP Cluster V95 to V95 versions of the cluster is also known as "compatibility mode". GBase 8a MPP Cluster V9.5 versions do

not support VC (virtual cluster), and upgrading to V95 version belongs to the compatibility mode of upgrading to V95. During the upgrade process, the cluster will automatically generate a default VC with vcname of vcname00000 1, and all nodes in the cluster belong to this VC by default.

In compatibility mode, cluster usage habits are the same as GBase 8a MPP Cluster V86 version, and VC related information cannot be seen using the gcadmin showcluster command.

The main installation and configuration steps for V95 single VC mode can be described as follows:

Execute gcinstall to install cluster software, → create distribution, → initialize cluster (initnodeDatamap)

3.2.3.1.1 Create distribution

Distribution determines the distribution pattern of data among various nodes in the cluster, including how many primary shards each data node stores, how many backup shards each primary shard has, and the pattern in which backup shards are distributed among cluster nodes. The detailed description of the various modes and parameters of data distribution is provided in the gcadmin tool. For detailed information, please refer to 4.3.1.1 Distribution Management Command. This section mainly demonstrates the complete steps of configuring a database.

Operating Steps

Step 1

After the cluster installation is successful, a gcChangeInfo.xml file will be generated in the installation package directory to describe the distribution of data among various nodes in the cluster. You can directly use the default gcChangeInfo.xml file in the gcinstall directory after installation to create a distribution, or you can use the dbaUser user (i.e. the dbaUser user in demo. options) to configure gcChangeInfo.xml as needed. The configuration of gcChangeInfo.xml for this example is as follows:

```
$cat /opt/gcinstall/gcChangeInfo.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
    <rack>
        <node ip="192.168.146.21"/>
        <node ip="192.168.146.41"/>
        <node ip="192.168.146.20"/>
        <node ip="192.168.146.40"/>
        <node ip="192.168.146.23"/>
        <node ip="192.168.146.43"/>
        <node ip="192.168.146.22"/>
        <node ip="192.168.146.42"/>
    </rack>
</servers>
```

Step 2

Execute the command to create a distribution.

```
gcadmin distribution <gcChangeInfo.xml> <p num> [d num] [pattern 1|2]
```

Table -39 Parameter Description

Parameter Name	Description
gcChangeInfo.xml	Specify the configuration file for generating distribution rules.
p number	The minimum number of shards stored in each data node is 1.
d number	The number of backups per shard, with values of 0, 1, or 2. If the parameter d is not entered, the default value is 1.
pattern number	The mode used to generate the distribution, with values of 1 or 2. Pattern 1 is the load balancing mode, and pattern 2 is the high availability mode. If the parameter pattern is not entered, the distribution will be generated using pattern 1 by default.

explain

The number of nodes in the rack in the gcChangeInfo.xml file needs to be greater than or equal to the value of parameter p (the number of primary shards stored in each node), otherwise an error will be reported as follows:

rack[1] node number:[1] shall be greater than segment number each node:[2]

Example: Generate Distribution

```
$ gcadmin distribution gcChangeInfo.xml p 2 d 1
gcadmin generate distribution ...
```

```
NOTE: node [192.168.146.21] is coordinator node, it shall be data node too
NOTE: node [192.168.146.20] is coordinator node, it shall be data node too
NOTE: node [192.168.146.22] is coordinator node, it shall be data node too
gcadmin generate distribution successful
```

3.2.3.1.2 Cluster initialization

Operation scenario

After the cluster is installed and the distribution is established, it is necessary to initialize the database system before executing SQL commands for the first time in order to perform all SQL operations correctly.

be careful

If the initialization operation is not performed, the database operation will prompt that it cannot be executed:

```
gbase> create database test;
ERROR 1707 (HY000): gcluster command error: (GBA-02CO-0003) nodedatamap is not initialized.
```

Operating Steps

Log in to the database using the database user root (the default password for the root user is blank) and execute the initnodedatmap command.

```
$ gccli -uroot

GBase client 9.5.3.17.123187. Copyright (c) 2004-2020, GBase.  All Rights Reserved.

gbase> initnodedatamap;
Query OK, 0 rows affected (Elapsed: 00:00:24.74)
```

be careful

This command only needs to be executed once. If executed repeatedly, the following information will be reported:

```
gbase> initnodedatamap;
ERROR 1707 (HY000): gcluster command error: (GBA-02CO-0004) nodedatamap is already initialized.
```

3.2.3.1.3 Post installation inspection

Operation scenario

After the cluster installation is completed, administrators can view the running status of the cluster through gcadmin.

prerequisite

The gcadmin command operates under the administrator user (i.e. the dbaUser specified during installation).

Operating Steps

Step 1

Check if the status of each node in the cluster is normal, and the displayed content is as follows:

```
$ gcadmin
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

=======================================
|   GBASE GCWARE CLUSTER INFORMATION   |
=======================================
| NodeName |      IpAddress       | gcware |
---------------------------------------
| gcware1    | 192.168.146.20 |    OPEN    |
---------------------------------------
| gcware2    | 192.168.146.21 |    OPEN    |
---------------------------------------
| gcware3    | 192.168.146.22 |    OPEN    |
---------------------------------------
=============================================================
|              GBASE COORDINATOR CLUSTER INFORMATION               |
=============================================================
|     NodeName     |      IpAddress       | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 192.168.146.20 |      OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 192.168.146.21 |      OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 192.168.146.22 |      OPEN     |        0        |
-------------------------------------------------------------
=============================================================
=======
|                              GBASE DATA CLUSTER INFORMATION
```

```
|
============================================================
=======
|NodeName|IpAddress |DistributionId |gnode |syncserver | DataState |
------------------------------------------------------------
|node1     |192.168.146.21   |    1     | OPEN   |       OPEN       |        0        |
------------------------------------------------------------
|node2      |192.168.146.41   |    1     | OPEN   |       OPEN       |        0        |
------------------------------------------------------------
|node3     |192.168.146.20   |    1     | OPEN   |       OPEN       |        0        |
------------------------------------------------------------
|node4     |192.168.146.40   | 1        | OPEN   |       OPEN       |        0        |
------------------------------------------------------------
|node5     |192.168.146.23   | 1        | OPEN   |       OPEN       |        0        |
------------------------------------------------------------
|node6     |192.168.146.43   |     1    | OPEN   |       OPEN       |        0        |
------------------------------------------------------------
|node7     |192.168.146.22   |     1    | OPEN   |       OPEN       |        0        |
------------------------------------------------------------
|node8     |192.168.146.42 |       1    | OPEN   |       OPEN       |        0        |
------------------------------------------------------------
```

Step 2

View the information related to cluster data sharding distribution, and the displayed content is as follows:

```
$ gcadmin showdistribution

Distribution ID: 1 | State: new | Total segment num: 16

Primary Segment Node IP      Segment ID       Duplicate Segment node IP
============================================================
=======
|          192.168.146.21         |       1       |       192.168.146.41                |
------------------------------------------------------------
|          192.168.146.41         |       2       |       192.168.146.20                |
------------------------------------------------------------
|          192.168.146.20         |       3       |         192.168.146.40              |
------------------------------------------------------------
|          192.168.146.40         |       4       |   192.168.146.23                    |
------------------------------------------------------------
|        192.168.146.23       |     5         |     192.168.146.43                    |
------------------------------------------------------------
|        192.168.146.43       |   6           |       192.168.146.22                  |
------------------------------------------------------------
|        192.168.146.22       |        7      |         192.168.146.42                |
------------------------------------------------------------
|          192.168.146.42     |        8      |         192.168.146.21                |
------------------------------------------------------------
|        192.168.146.21       |       9       |         192.168.146.20                |
------------------------------------------------------------
|      192.168.146.41         |        10     |       192.168.146.40                  |
```

```
-------------------------------------------------------------
|     192.168.146.20      |      11   |      192.168.146.23          |
-------------------------------------------------------------
|     192.168.146.40      |      12   |      192.168.146.43          |
-------------------------------------------------------------
|     192.168.146.23      |      13   |      192.168.146.22          |
-------------------------------------------------------------
|     192.168.146.43      |      14   |      192.168.146.42          |
-------------------------------------------------------------
|     192.168.146.22      |      15 |      192.168.146.21          |
-------------------------------------------------------------
|     192.168.146.42      |      16 |      192.168.146.41          |
=============================================================
=======
```

Operation results

The inspection results show that the status of each node is "OPEN" and can be used normally in the future.

explain

- For more gcadmin subcommands, please refer to the section 4.3.1 gcadmin.

3.2.3.2 Initial configuration of multi VC mode

Related concepts

Multi VC mode "refers to the cluster mode where data nodes are divided into multiple VCs using the gcadmin createvc command after installing the cluster. At this point, using the gcadmin command can display information about multiple VCs. In the multi VC mode, it is necessary to first create a VC and then create a distribution on each VC.

explain

The example explanation in this section is as follows:

- In this example, there are 8 free nodes, with IP addresses of 192.168.146.20192.168.146.21192.168.146.22192.168.146.23192.168.146.40192.168.146.41192.168.146.42192.168.146.43, respectively,;
- VC1 contains 4 data nodes: 192.168.146.20192.168.146.40192.168.146.21192.168.146.41;
- VC2 contains 4 data nodes: 192.168.146.22192.168.146.42192.168.146.23192.168.146.43.

3.2.3.2.1 Create VC1

Operation scenario

Use multi VC mode.

prerequisite

The cluster is running normally;

There is a freenode in the root cluster;

Gcadmin is available.

Operating Steps

步骤 1

Switch to the dbaUser user (i.e. dbaUser in demo. options) and enter the gcinstall directory after unpacking the installation package.

```
# su - gbase
$ cd /opt/gcinstall
```

步骤 2

Generate an example configuration demonstration file for creating VC1, generated by the following command.

```
$ gcadmin createvc e example_ file_ name
```

Among them: example_ file_ Name is the generated file, which can be any name, such as' create '_ vc.xml。

The generated configuration file command reference is as follows:

```
$ gcadmin createvc e create_ vc1.xml
$ cat create_ vc1.xml
<? xml version='1.0' encoding="utf-8"?>
<servers>

    <rack>
        <node ip="vc data node ip"/>
        <!--- ... -->
        <node ip="vc data node ip"/>
```

```
    </rack>

    <vc_ name name="virtual cluster name no more than 64 bytes"/>
    <comment message="comment message no more than 60 bytes"/>

</servers>
```

explain

- vc_ The name cannot exceed 64 bytes, and the comment information cannot exceed 60 bytes;
- The node IP in the configuration file for creating VC is the IP of Free node, and cannot be shared among multiple VCs;

步骤 3

Edit the configuration file for creating VC1.

Example: Editing create_ The content of vc1. xml is as follows:

```
$ vi create_ vc1.xml
<? xml version='1.0' encoding="utf-8"?>
<servers>

    <rack>
        <node ip="192.168.146.20"/>
        <node ip="192.168.146.40"/>
        <node ip="192.168.146.21"/>
        <node ip="192.168.146.41"/>
    </rack>

    <vc_ name name="vc1"/>
    <comment message="vc1comments"/>

</servers>
(: wq save exit)
```

步骤 4

Execute the create command.

```
$ gcadmin createvc create_ vc1.xml
```

The execution results are as follows:

```
$ gcadmin createvc create_ vc1.xml
parse config file create_ vc1.xml
generate vc id: vc00001
add vc information to cluster
add nodes to vc
gcadmin create vc [vc1] successful
```

3.2.3.2.2 Create distribution on VC1

Operation scenario

Multi VC mode, distribution needs to be created under each VC.

Operating Steps

步骤 1

Edit the configuration file gcChangeInfo for creating distribution for vc1_ vc1.xml。

When creating a distribution in the specified VC, create the configuration file gcChangeInfo for the distribution_ The node IP in vc1. xml is consistent with the node IP in the configuration file where the VC was created.

Example: Editing the configuration file gcChangeInfo for creating a distribution on vc1_ The content of vc1. xml is as follows:

```
$cd gcinstall
$cp gcChangeInfo.xml gcChangeInfo_ vc1.xml
$vi /opt/gcinstall/gcChangeInfo_ vc1.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
    <rack>
        <node ip="192.168.146.21"/>
        <node ip="192.168.146.41"/>
    </rack>
    <rack>
        <node ip="192.168.146.20"/>
        <node ip="192.168.146.40"/>
    </rack>
</servers>
```

步骤 2

Execute the create distribution command in the installation directory.

```
gcadmin distribution <gcChangeInfo.xml> <p num> [d num] [pattern 1|2] <vc vcname>
```

Table 3-10 Parameter Description

Parameter Name	Description
gcChangeInfo.xml	Specify the configuration file for generating distribution rules.
p number	The minimum number of shards stored in each data node is 1.
d number	The number of backups per shard, with values of 0, 1, or 2. If the parameter d is not entered, the default value is 1.
pattern number	The mode used to generate the distribution, with values of 1 or 2. Pattern 1 is the load balancing mode, and pattern 2 is the high availability mode. If the parameter pattern is not entered, the distribution will be generated using pattern 1 by default.
vc vcname	The VC name to create the distribution.

explain

- Gcadmin distribution is a subcommand of gcadmin. For detailed instructions, please refer to the section 4.3.1 gcadmin.

The execution results are as follows:

```
$cd /opt/gcinstall
$ gcadmin distribution gcChangeInfo_ vc1.xml p 1 d 1 vc vc1
gcadmin generate distribution ...

NOTE: node [192.168.146.21] is coordinator node, it shall be data node too
NOTE: node [192.168.146.20] is coordinator node, it shall be data node too
gcadmin generate distribution successful
```

explain

- When creating a distribution on a VC, gcChangeInfo_ The node IP in the VC1. XML file should be consistent with the node IP in the configuration file when creating the VC;
- When creating a distribution, the VC name must be specified. When the cluster has only one VC, the distribution can be generated on that VC by default without specifying the VC name.

3.2.3.2.3 Create distribution for VC2 and VC2

Operation scenario

Use multi VC mode.

prerequisite

- The cluster is running normally;
- There is a freenode in the root cluster;
- Gcadmin is available.

Operating Steps

步骤 1

Switch to the dbaUser user (i.e. dbaUser in demo. options), enter the gcinstall directory after unpacking the installation package, and generate an example configuration demonstration file for creating VC2.

Example:

```
# su - gbase
$ cd /opt/gcinstall
$ gcadmin createvc e create_ vc2.xml
$ cat create_ vc2.xml
<? xml version='1.0' encoding="utf-8"?>
<servers>

    <rack>
        <node ip="192.168.146.22"/>
        <node ip="192.168.146.42"/>
        <node ip="192.168.146.23"/>
        <node ip="192.168.146.43"/>
    </rack>

    <vc_ name name="vc2"/>
    <comment message="vc2comments"/>

</servers>
(: wq save exit)
```

 **explain**

- vc_ The name cannot exceed 64 bytes, and the comment information cannot exceed 60 bytes;
- The node IP in the configuration file for creating VC is the IP of Free node, and cannot be shared among multiple VCs;

步骤 2

Execute the create vc2 command.

```
$ gcadmin createvc create_ vc2.xml
```

The execution results are as follows:

```
$ gcadmin createvc create_ vc2.xml
parse config file create_ vc2.xml
generate vc id: vc00002
add vc information to cluster
add nodes to vc
gcadmin create vc [vc2] successful
```

步骤 3

Edit the configuration file gcChangeInfo for creating distribution on vc2_ VC2. XML content.

Example:

```
$cd gcinstall
$cp gcChangeInfo.xml gcChangeInfo_ vc2.xml
$vi /opt/gcinstall/gcChangeInfo_ vc2.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
    <rack>
        <node ip="192.168.146.22"/>
        <node ip="192.168.146.42"/>
    </rack>
    <rack>
        <node ip="192.168.146.23"/>
        <node ip="192.168.146.43"/>
    </rack>
</servers>
```

步骤 4

Execute the create distribution command in the installation directory.

The execution results are as follows:

```
$ gcadmin distribution gcChangeInfo_ vc2.xml p 1 d 1 db_ user user_ name db_ pwd password vc vc2
gcadmin generate distribution ...

NOTE: node [192.168.146.22] is coordinator node, it shall be data node too
gcadmin generate distribution successful
```

3.2.3.2.4 Cluster initialization

Operation scenario

Each VC must first execute initnodeDatamap after creating a distribution.

Operating Steps

Log in as the database user root (the default password for the root user is blank) and execute the initnodeDatamap command.

Example: Execute initnodeDatamap on vc1 and vc2 that have already been created.

```
$ gccli -uroot

GBase client 9.5.3.17.123187. Copyright (c) 2004-2020, GBase.    All Rights Reserved.

gbase> use vc vc1;
Query OK, 0 rows affected (Elapsed: 00:00:00.04)

gbase> initnodedatamap;
Query OK, 0 rows affected (Elapsed: 00:00:10.83)

gbase> use vc vc2;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)

gbase> initnodedatamap;
Query OK, 0 rows affected (Elapsed: 00:00:15.78)
```

3.2.3.2.5 **Post installation inspection**

Operation scenario

After the cluster installation is completed, administrators can view the running status of the cluster through gcadmin.

prerequisite

The gcadmin command operates under the administrator user (i.e. the dbaUser specified during installation).

Operating Steps

Step 1

Execute the gcadmin command under the administrator user to check if the status of each node in the cluster is normal. The displayed content is as follows:

```
$ gcadmin
CLUSTER STATE:              ACTIVE


=====================================
|   GBASE GCWARE CLUSTER INFORMATION   |
=====================================
| NodeName |     IpAddress      | gcware |
---------------------------------------
| gcware1   | 192.168.146.20 |   OPEN   |
---------------------------------------
| gcware2   | 192.168.146.21 |   OPEN   |
---------------------------------------
| gcware3   | 192.168.146.22 |   OPEN   |
---------------------------------------
=============================================================
|              GBASE COORDINATOR CLUSTER INFORMATION              |
=============================================================
|     NodeName     |     IpAddress      | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 192.168.146.20 |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 192.168.146.21 |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 192.168.146.22 |     OPEN     |        0        |
```

```
-----------------------------------------------------
=============================================
|        GBASE VIRTUAL CLUSTER INFORMATION         |
=============================================
|       VcName       | DistributionId |    comment    |
---------------------------------------------
|         vc1          |            1             | comment     |
---------------------------------------------
|         vc2          |            2             | comment     |
---------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node


$ gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

=======================================
|     GBASE VIRTUAL CLUSTER INFORMATION     |
=======================================
| VcName | DistributionId |    comment       |
-----------------------------------------
|   vc1     |            1             | vc1comments |
-----------------------------------------
=============================================================
=======
|                          VIRTUAL  CLUSTER  DATA  NODE  INFORMATION
|
=============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-------------------------------------------------------------
| node1    |192.168.146.20|       1              |OPEN |    OPEN       |     0        |
-------------------------------------------------------------
| node2    |172.168.146.40|       1              |OPEN |    OPEN       |     0        |
-------------------------------------------------------------
| node1    |192.168.146.21|       1              |OPEN |    OPEN       |     0        |
-------------------------------------------------------------
| node2    |172.168.146.41|       1              |OPEN |    OPEN       |     0        |
-------------------------------------------------------------

4 data node

$ gcadmin showcluster vc vc2
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL
```

```
==========================================
|      GBASE VIRTUAL CLUSTER INFORMATION    |
==========================================
| VcName |DistributionId |      comment       |
------------------------------------------
|   vc2    |       2                | vc2comments    |
------------------------------------------
================================================================
=======
|                              VIRTUAL CLUSTER DATA NODE INFORMATION
|
================================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
----------------------------------------------------------------
| node1    |192.168.146.22|       1              |OPEN |    OPEN       |     0        |
----------------------------------------------------------------
| node2    |172.168.146.42|       1              |OPEN |    OPEN       |     0        |
----------------------------------------------------------------
| node1    |192.168.146.23|       1              |OPEN |    OPEN       |     0        |
----------------------------------------------------------------
| node2    |172.168.146.43|       1              |OPEN |    OPEN       |     0        |
----------------------------------------------------------------

4 data node
```

Step 2

View information related to cluster data sharding distribution. The displayed content is as follows:

```
$gcadmin showdistribution vc vc1

            Distribution ID: 1 | State: new | Total segment num: 4

 Primary Segment Node IP     Segment ID     Duplicate Segment node IP
================================================================
=======
|     192.168.146.21         |         1          |        192.168.146.40           |
----------------------------------------------------------------
|     192.168.146.41         |         2          |        192.168.146.20           |
----------------------------------------------------------------
|     192.168.146.20         |         3          |        192.168.146.41           |
----------------------------------------------------------------
|     192.168.146.40         |         4          |        192.168.146.21           |
================================================================
=======
$gcadmin showdistribution vc vc2

            Distribution ID: 2 | State: new | Total segment num: 4

 Primary Segment Node IP    Segment ID       Duplicate Segment node IP
================================================================
=======
```

```
|     192.168.146.23      |      1      |     192.168.146.42      |
-----------------------------------------------------------------
|     192.168.146.43      |      2      |     192.168.146.22      |
-----------------------------------------------------------------
|     192.168.146.22      |      3      |     192.168.146.43      |
-----------------------------------------------------------------
|     192.168.146.42      |      4      |     192.168.146.23      |
=================================================================
=======
```

Operation results

The inspection results show that the status of each node is "OPEN" and can be used normally in the future.

explain

- For more gcadmin subcommands, please refer to the section 4.3.1 gcadmin.
- For more VC management commands, please refer to the section 4.3.2 VC Management.

3.2.4 Full text retrieval function installation

Please refer to the following content for the installation steps of the full-text search function:

Step 1

The cluster product has been successfully installed and all cluster services have been stopped.

DBA (gbase) users execute the following command on all nodes of the cluster to stop cluster services:

```
$ gcluster_ services all stop
Stopping gcrecover :                                    [  OK  ]
Stopping gcluster :                                     [  OK  ]
Stopping gbase :                                        [  OK  ]
Stopping syncserver :                                   [  OK  ]

$ gcware_ services all stop
Stopping GCWareMonit success!
Stopping gcware :                                       [  OK  ]
```

Step 2

Switch to the cluster installation user, copy the installation package to the directory of the file system, enter the directory, and extract the installation package.

The reference commands are as follows:

```
$cp /tmp/GBase8a_ MPP_ Cluster-NoLicense-fulltext-9.5.3.17-redhat7.3-x86_64.tar.bz2 /opt
$cd /opt
$ tar xjf GBase8a_ MPP_ Cluster-NoLicense-fulltext-9.5.3.17-redhat7.3-x86_64.tar.bz2
```

Step 3

Execute installation command:

```
# ./ gcinstall_ fulltext.py < --dbaUserPwd=DBAPWD>
[--passwordInputMode=PASSWORDINPUTMODE]
```

Table 315 Parameter Description

Parameter Name	Description
dbaUserPwd	Specify the password of the cluster DBA user.
PasswordInputMode	Optional parameters, specifying the method of obtaining passwords, and implementing different methods of obtaining passwords through different parameters. If this parameter is specified, the password in demo. options does not need to be modified again. The value range is [file, pwdsame, pwddiff], and the default value is file: ● File: represents obtaining from a file, in which the password in the file is plaintext; ● Pwdsame: This parameter is used when the password is entered by the user from the terminal and the passwords of all nodes are consistent. For different user passwords, only one input is required; ● Pwddiff: This parameter is used when the password is entered by the user from the terminal and the passwords between nodes are inconsistent. For different user passwords, each node is input once.

Example:

```
$ cd /opt/gcinstall_ fulltext
$./ gcinstall_ fulltext.py --dbaUserPwd=gbase
CoordinateHost:
172.168.83.11    172.168.83.12    172.168.83.13
DataHost:
172.168.83.11    172.168.83.12    172.168.83.13    172.168.83.14
Are you sure to install fulltext on these gcluster nodes ([Y,y]/[N,n])? y
172.168.83.11                Install fulltext successfully.
```

```
172.168.83.12        Install fulltext successfully.
172.168.83.13        Install fulltext successfully.
172.168.83.14        Install fulltext successfully.
```

Step 4

Execute the following command on all nodes of the cluster to start the cluster service:

```
$ gcluster_ services all start
Starting gbase :                                    [  OK  ]
Starting syncserver :                               [  OK  ]
Starting gcluster :                                 [  OK  ]
Starting gcrecover :                                [  OK  ]
$ gcware_ services all start
Starting gcware :                                   [  OK  ]
Starting GCWareMonit success!
```

After installation, please refer to the full-text index chapter for usage.

3.2.5 Install client (optional)

Operation scenario

This operation guides engineers to install the client of GBase 8a MPP Cluster service.

prerequisite

1. The cluster client installation tool can be independently installed on a machine in a non cluster environment, which is interconnected with the cluster business network;
2. Consistent with the Linux operating system supported by GBase 8a MPP Cluster.

Operating Steps

步骤 1

Obtain the software package and extract it. You can use any operating system user for decompression, and after decompression, gccli will be generated in the decompression directory_ Install directory.

The software installation package version rules are as follows:

```
gccli-9.5.3.xx-OSversion-platform.tar.bz2
```

The decompression command is as follows:

```
$ tar xjf gccli-9.5.3.17-redhat7.3-x86_ 64.tar.bz2
```

步骤 2

Perform installation.

Copy the extracted folder gccli_ Install the content to a custom installation path, and execute the installation program under the installation path.

Assuming installation path/home/test/gccli_ Client, enter the installation path and execute the installation command.

```
$ chmod 744 gccli_ install.sh

$./ gccli_ install.sh gccli_ standalone.tar.bz2
```

After successful installation, a gcluster directory will be generated in the installation path. The screen displays as follows:

```
gcluster/
gcluster/server/
gcluster/server/lib/
gcluster/server/lib/gbase/
gcluster/server/bin/
gcluster/server/bin/gbase
gcluster/server/bin/gcdump
gcluster/config/
gcluster/config/gbase_ 8a_ gcluster.cnf
Installation finished.
Please run "/home/test/gccli_client/gcluster/server/bin/gccli -uUSER -pPASSWORD -hGCLUSTER_NODE_IP" for checking.
```

步骤 3

Check if the client installation was successful.

Enter the client installation directory and log in to the cluster according to the prompt message on the last line of successful installation in step 2.

Example:

```
$ /home/test/gccli_ client/gcluster/server/bin/gccli -uroot –p****** -h

192.168.105.100
```

Table -316 Parameter Description

Parameter Name	Description
-h	Specify the connection IP, which must be the coordinator cluster node of the cluster;
-u	Specify database user
-p	Represents the password of the database user specified by the - u parameter.

3.2.6 Port modification

3.2.6.1 GCLuster Port Modification

Example: To modify gcluster service port 5258 to 5259, the following operations need to be performed on all nodes of the cluster:

Operating Steps

步骤 1

Modifying $GCLUSTER using DBAUSER users_ BASE/config/gbase_ 8a_ The gcluster.cnf file changes the 5258 in the configuration file to 5259. The modification command reference is as follows:

```
# su - gbase
$ vi $GCLUSTER_ BASE/config/gbase_ 8a_ gcluster.cnf
[client]
port=5259
socket=/tmp/gcluster_ 5259.sock
……
[gbased]
……
socket=/tmp/gcluster_ 5259.sock
……
port=5259
……
```

步骤 2

Modify the gcware. conf configuration file:

```
$vi $GCWARE_ BASE/config/gcware.conf
totem {
……
}
logging {
……
}
```

```
gcware {
……
      gcluster_ port: 5259
……
}
```

步骤 3

After modifying the configuration files on each node, restart the cluster service:

```
$ gcluster_ services all start
Starting gbase :                                    [  OK  ]
Starting syncserver :                               [  OK  ]
Starting gcluster :                                 [  OK  ]
Starting gcrecover :                                [  OK  ]
$ gcware_ services all start
Starting gcware :                                   [  OK  ]

Starting GCWareMonit success!
```

步骤 4

Verify if the new port was successfully modified:

Method 1: Use cluster command line mode login verification:

```
$ gccli -uroot -h172.168.83.11 -P5259


GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.   All Rights
Reserved.


gbase> quit
Bye
```

Method 2: Verify using operating system commands:

```
$ lsof -i:5259
COMMAND     PID   USER    FD     TYPE DEVICE SIZE/OFF NODE
NAME
gclusterd 4780 gbase    14u    IPv4   23640       0t0    TCP *:5259 (LISTEN)
```

3.2.6.2 Gnode port modification

Example: To modify the gnode service port 5050 to 5051, the following operations need t o be performed on all nodes of the cluster:

Operating Steps

步骤 1

Modifying $GCLUSTER using DBAUSER users_ BASE/config/gbase_ 8a_ The gcluster.cnf file changes 5050 in the configuration file to 5051. The modification command reference is as follows:

```
# su - gbase
$ vi $GCLUSTER_ BASE/config/gbase_ 8a_ gcluster.cnf
[client]
……
[gbased]
……
gcluster_ gnode_ port = 5051
……
```

步骤 2

Using DBAUSER Users to Modify $GBASE_ BASE/config/gbase_ 8a_ The specific modified parameters for the relevant content in the gbase.cnf file are as follows:

```
$ vi    $GBASE_ BASE/config/gbase_ 8a_ gbase.cnf
[client]
port=5051
socket=/tmp/gbase_ 8a_ 5051.sock

[gbased]
……
socket=/tmp/gbase_ 8a_ 5051.sock
……
port=5051
……
```

步骤 3

Modifying $GCWARE using DBAUSER users_ The specific parameters modified in the BASE/config/gcware. conf file are as follows:

```
$vi $GCWARE_ BASE/config/gcware.conf
totem {
……
}
logging {
……
}

gcware {
……
      gnode_ port: 5051
……
}
```

步骤 4

Modifying $GCLUSTER using DBAUSER users_ BASE/config/gc_ The specific parameters modified in the recover.cnf file are as follows:

```
$vi $GCLUSTER_ BASE/config/gc_ recover.cnf
[DDL Recovery]
……
[DATA Recovery]
……
recover_ monit_ port = 6268
gcluster_ gnode_ port=5051
```

步骤 5

After modifying the configuration files on each node, restart the cluster service:

```
$ gcluster_ services all start
Starting gbase :                                    [  OK  ]
Starting syncserver :                               [  OK  ]
Starting gcluster :                                 [  OK  ]
Starting gcrecover :                                [  OK  ]
$ gcware_ services all start
Starting gcware :                                   [  OK  ]
Starting GCWareMonit success!
```

步骤 6

Verify if the new port was successfully modified:

```
$ lsof -i:5051
COMMAND   PID   USER    FD    TYPE DEVICE SIZE/OFF NODE NAME
gbased  3452 gbase     5u  IPv4  17730       0t0  TCP *5051(LISTEN)
```

3.2.6.3 Syncserver Port Modification

Example: To modify the syncserver service port 5288 to 5287, the following operations need to be performed on all nodes of the cluster:

Operating Steps

步骤 1

Using DBAUSER Users to Modify $GBASE_ The BASE/config/synctool. conf file changes the 5288 in the configuration file to 5287. The modification command reference is as follows:

```
$ vi $GBASE_ BASE/config/synctool.conf
SERVER_ PORT=5287
……
```

步骤 2

Modifying $GCWARE using DBAUSER users_ The specific parameters modified in the BASE/config/gcware. conf file are as follows:

```
$vi $GCWARE_ BASE/config/gcware.conf
totem {
……
}
logging {
……
}

gcware {
……
    syncserver_ port: 5287
……
}
```

步骤 3

After modifying the configuration files on each node, restart the cluster service:

```
$ gcluster_ services all start
Starting gbase :                                    [  OK  ]
Starting syncserver :                               [  OK  ]
Starting gcluster :                                 [  OK  ]
Starting gcrecover :                                [  OK  ]
$ gcware_ services all start
Starting gcware :                                   [  OK  ]
Starting GCWareMonit success!
```

步骤 4

Verify if the new port was successfully modified:

```
$ lsof -i:5287
COMMAND      PID   USER    FD     TYPE DEVICE SIZE/OFF NODE
NAME
gc_ sync_ s 3876 gbase     4u   IPv4  20103       0t0   TCP *:5287 (LISTEN)
```

3.3 Cluster multi instance installation and deployment

3.3.1 Suggestions for multi instance deployment

Operation scenario

Multi instance deployment is only supported in V9.5.3. Deploying multiple data cluster nodes on a physical server, each of which is referred to as a database instance.

When GBase 8a MPP Cluster is deployed on high-end servers (usually using non unified memory access architecture, referred to as NUMA architecture), such as memory greater than 256GB and CPU logical cores greater than 32, it improves cluster performance by deploying multiple database instances on a single server. GBase 8a MPP Cluster installs and deploys multiple data nodes on each server. Each data node has an independent IP address, which is used to distinguish between different nodes. At most one gcluster node and one gcware node can be deployed on each physical server.

Deployment suggestions

- It is recommended to apply for an independent IP address for each data node.
- It is recommended that the IP addresses of multiple data nodes on the same server should be as discontinuous as possible, in order to avoid the default of multiple consecutive IP data nodes fetching data during loading, resulting in pressure concentration on some servers and causing a barrel effect due to excessive pressure on some servers.
- It is recommended to have an odd number of gcware nodes and gccluster nodes, and only one gcware node and one gccluster node can be deployed on each physical server. Both the gcware cluster and the gccluster cluster can provide normal services to the outside world when more than half of the nodes are normal, so odd number of nodes are usually deployed, and it is not allowed to deploy more than one node on a server.
- Multi instance deployment can treat each physical server as a rack and generate distributions in a highly available manner according to the rack, thus avoiding the primary and backup data shards of the table being located on the same physical machine.

  For example: Server 1: 172.16.3.61, 172.16.3.64

  Server 2: 172.16.3.62, 172.16.3.65

	Pattern 1 Backup to other rack racks	Pattern 2 Backup to adjacent nodes

	Pattern 1 Backup to other rack racks	Pattern 2 Backup to adjacent nodes
distribution gcChangeInfo.xml	<servers> <rack> <node ip="172.16.3.61"/> <node ip="172.16.3.64"/> </rack> <rack> <node ip="172.16.3.62"/> <node ip="172.16.3.65"/> </rack> </servers>	<servers> <rack> <node ip="172.16.3.61"/> <node ip="172.16.3.62"/> <node ip="172.16.3.64"/> <node ip="172.16.3.65"/> </rack> </servers>

- Evaluate the number of machine nodes deployed on each server based on the number of numa nodes, memory size, cluster size, and business scenario (load). It is recommended to deploy no more than 4 instances per server, and each instance can use no less than 32GB of memory. On servers with 4 NUMA nodes, each NUMA corresponds to an instance; On servers with 8 or more NUMA nodes, 2 or more NUMA nodes correspond to one instance.
- It is recommended to deploy the gcware node and gcluster node on the same numa node, not together with the data node.

 be careful

Edit gcChangeInfo_ When creating the vcall.xml file, it is necessary to backup each IP shard of the same server to another server.

If using pattern 1 to create a distribution, it is necessary to place the IP of the same server on a single rack, so that all backup shards fall on adjacent racks, ensuring that the backup and primary shards fall on different servers;

If using pattern 2 to create a distribution, as this mode places backup partitions on adjacent IPs, it is necessary to store the IPs of different servers in gcChangeInfo_ Adjacent emissions in vcall.xml.

3.3.2 Multiple instance license acquisition

The steps and methods for obtaining the license files required for cluster multi instance deployment are the same as those for regular cluster deployment:

1. Generate ESN on cluster servers in advance
2. Send ESN and license restriction requirements to license@gbase.cn
3. Obtain license file
4. Execute the cluster installation command to install the cluster normally

For detailed steps, please refer to 3.2.1 to obtain the license section

3.3.3 Multiple instance installation and deployment

3.3.3.1 Step 1 Environmental Preparation

- IP preparation

A server equipped with multiple 10 Gigabit network cards can be assigned different IP addresses on different network cards.

When there is only one network card or the number of network cards is less than the number of IP addresses, IP addresses can be set for different nodes by configuring virtual IP addresses.

The requirements for other environments are consistent with those for non multi instance regular installation environments. You can refer to 3.1 Installation Environment Preparation.

```
Vim/etc/sysconfig/network scripts/ifcfg-p6p2 (please change to the file name corresponding to the network card)
TYPE=Ethernet
BOOTPROTO=none
NAME=enp49s0f0
UUID=f3004479-00cb-4593-a7bc-50af8d9c27f6
DEVICE=enp49s0f0
ONBOOT=yes
IPADDR=192.168.146.10
NETMASK=255.255.255.0
GATEWAY=192.168.146.254
IPADDR1=192.168.146.20
NETMASK1=255.255.255.0
IPADDR2=192.168.146.30
```

```
NETMASK2=255.255.255.0
IPADDR3=192.168.146.40
NETMASK3=255.255.255.0
```

3.3.3.2 Step 2 Initial installation of multiple instances

The installation example IP environment in this section is as follows:

Server 1: 172.16.3.61, 172.16.3.64

Server 2: 172.16.3.62, 172.16.3.65

- The initial installation steps for the cluster are the same as for non multi instance regular installation, please refer to 3.2.2 Initial Installation.

Step 1: Create database DBA users on each server in the cluster, such as gbase users;

```
# useradd gbase
# passwd gbase
```

Step 2: Root user execution: Obtain the cluster installation package and extract it; Set the owner of the decompression directory gcinstall and the installation directory where the cluster will be installed to DBA user gbase; Copy the SetSysEnv.py file from the extracted directory gcinstall to each cluster server and execute it (each server only needs to copy and execute any IP address once).

```
#cd /opt
# tar xjf  GBase8a_ MPP_ Cluster-License-9.5.3.27-redhat7.3-x86_
64.tar.bz2
# chown –R gbase:gbase gcinstall
# chown –R gbase:gbase /opt
# scp /opt/gcinstall/SetSysEnv.py  root@192.168.146.61 :/opt
# scp /opt/gcinstall/SetSysEnv.py  root@192.168.146.62 :/opt
#Log in to two servers, 192.168.146.61 and 192.168.146.62, and execute:
# /opt/SetSysEnv.py  --dbaUser=gbase --installPrefix=/opt
```

Step 3: DBA user gbase execution: Modify the demo.options file in the gcinstall directory, write the IP addresses of all planned instances into the dataHost parameter, and execute gcinstall.py to install the cluster.

```
$ cat demo.options
```

```
installPrefix= /opt
coordinateHost = 172.16.3.61,172.16.3.62
coordinateHostNodeID = 61,62
dataHost = 172.16.3.61,172.16.3.64,172.16.3.62,172.16.3.65
#existCoordinateHost =
#existDataHost =
#existGcwareHost=
gcwareHost = 172.16.3.61,172.16.3.62
#gcwareHostNodeID =
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbase'
rootPwd = '222222'
#rootPwdFile = rootPwd.json
#characterSet = utf8
#dbPort = 5258
#sshPort = 22
GBase User Execution
./gcinstall.py --silent=demo.options --license_ file=20210323.lic
```

Step 4: Each server executes the environment configuration script again:

```
# /opt/SetSysEnv.py    --dbaUser=gbase --installPrefix=/opt
```

3.3.3.3 Step 3 Initial configuration of multiple instances

The initial configuration of multiple instances can use either single VC mode or multi VC mode. The operation and configuration are the same as for non multi instance regular installations. Please refer to 3.2.2 Initial Configuration.

Multiple instances suggest that all data nodes on the same server belong to the same VC. When configuring VC in multi VC mode, it is important to place all IPs of the same server within the same VC. For example:

Step 1: Create VC

```
vi vc.xml
    <? xml version='1.0' encoding="utf-8"?>
```

```
    <servers>
        <rack>
            <node ip="172.16.3.61"/>
            <node ip="172.16.3.62"/>
            <node ip="172.16.3.64"/>
            <node ip="172.16.3.65"/>
        </rack>
        <vc_ name name="vc1"/>
        <comment message="vc1"/>
</servers>
gcadmin createvc    vc.xml
```

Step 2: Create a distribution

Multiple instances need to ensure that the master data on the same server is sharded and backed up to other servers. In this example, the default pattern 1 is used to place data nodes on the same server on the same rack. As follows:

```
Modify gcChangeInfo.xml to place the IP of the same server in a rack.
$ vi gcChangeInfo.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
    <rack>
        <node ip="172.16.3.61"/>
        <node ip="172.16.3.64"/>
    </rack>
    <rack>
        <node ip="172.16.3.62"/>
        <node ip="172.16.3.65"/>
    </rack>
</servers>
gcadmin distribution gcChangeInfo.xml p 1 d 1
Generate a new hashmap
$ gccli
gbase> use vc vc1;
gbase> initnodedatamap;
```

3.3.4 Multi instance NUMA binding

- summary:

The command to start all instances on a multi instance server is as follows, which randomly points to the gcluster of any instance on the server during execution_ Services script.

```
$ gcluster_ services all start
```

Gcluster on cluster servers_ There will be one or more services scripts, and the content of the scripts will be identical by default. They exist in the server/bin directory of the gcluster directory and in the server/bin directory of all instances of gnode. In this example, there is a gcluster on server one_ The directory for the services script is as follows:

```
In this example, the installation directory is/opt, and/opt below represents the cluster installation directory
$GCLUSTER_ BASE/server/bin/gcluster_ services
Gcluster under all gnode instances_ services
/opt/172.16.3.61/gnode/server/bin/gcluster_ services
/opt/172.16.3.64/gnode/server/bin/gcluster_ services
```

Multiple instance binding numas requires determining how many numas correspond to one instance based on information such as the number of CPU cores, memory size, and the number of instances deployed on the physical machine of the server, and then starting the service script gcluster in the cluster_ Modify the configuration in services to make NUMA binding. After NUMA binding, it is necessary to use a modified gcluster_ The services script restarts the cluster service to make NUMA binding effective. All subsequent startup and shutdown of cluster services on this server require the use of this gcluster_ Services script to ensure NUMA binding takes effect.

NUMA binding suggests specifying gcluster under fixed instance gnode/server/bin_ Add binding commands to the services file, and use this file to start all subsequent database services. For example:

```
cd 172.16.3.61/gnode/server/bin
./gcluster_ services all start
```

- NUMA binding preparation

Check hardware numa support:

```
# dmesg | grep -i numa
[    0.000000] Enabling automatic NUMA balancing. Configure with numa_
balancing= or the kernel.numa_ balancing sysctl
[    0.731820] pci_ bus 0000:00: on NUMA node 0
[    0.735153] pci_ bus 0000:40: on NUMA node 1
[    0.737492] pci_ bus 0000:3f: on NUMA node 0
[    0.739961] pci_ bus 0000:7f: on NUMA node 1
```

Check the operating system numactl tool:

```
# numactl --hardware
available: 2 nodes (0-1)
node 0 cpus: 0 2 4 6 8 10 12 14
node 0 size: 32722 MB
node 0 free: 22537 MB
node 1 cpus: 1 3 5 7 9 11 13 15
node 1 size: 32768 MB
node 1 free: 23142 MB
node distances:
node   0   1
  0:  10  20
  1:  20  10
```

If the hardware supports numa and the operating system does not install numa tools, a separate installation of numactl tools is required:

yum install -y numactl
Alternatively, use the rpm command to install the following package (if it is an arm architecture, please replace x86):
numactl-2.0.9-7.el7.x86_ 64.rpm
numactl-devel-2.0.9-7.el7.x86_ 64.rpm
numactl-libs-2.0.9-7.el7.x86_ 64.rpm
numad-0.5-18.20150602git.el7.x86_ 64.rpm

After installation, you can view the current server numa status:

```
# numastat
                           node0           node1
numa_ hit                52237686       241164821
numa_ miss                     0               0
numa_ foreign                  0               0
```

interleave_ hit	22797	22887
local_ node	52226319	241150686
other_ node	11367	14135

- Binding steps

NUMA binding requires modifying gcluster_ After two services scripts, use this gcluster_ The services script restarts the cluster service.

As specified in the 172.16.3.61 instance of server 1, gcluster_ Services script for numa binding

The first modification is as follows:

```
$ cd /opt/172.16.3.61/gnode/server/bin
$ vi gcluster_ services
Find the following code around 410 lines
$2 > /dev/null 2>&1 &
# waiting for start completely
Modify as follows and add Code Red
$2 > /dev/null 2>&1 &
        # echo "$prog_name ----------------------$2"
          if [ $prog_name = '/opt/172.16.3.61/gnode/server/bin/gbased' ]; then
                #echo -e "\n------numactl${loop_count}------$3"
                count_ numa=$(($(($3))%2))
                echo    "--cpunodebind=+$count_numa --membind=+$count_numa"
                numactl --cpunodebind=+"${count_numa}"
--membind=+"${count_numa}" $2 > /dev/null 2>&1 &
                                #                       $2 > /dev/null 2>&1 &
                sleep 10
              else
                        $2 > /dev/null 2>&1 &
              fi

        if [ $prog_name = '/opt/172.16.3.64/gnode/server/bin/gbased' ]; then
                #echo -e "\n------numactl${loop_count}------$3"
                count_ numa=$(($(($3))%2))
                echo    "--cpunodebind=+$count_numa --membind=+$count_numa"
```

```
                numactl --cpunodebind=+"${count_numa}"
--membind=+"${count_numa}" $2 > /dev/null 2>&1 &
                        #                       $2 > /dev/null 2>&1 &
                sleep 10
            else
                    $2 > /dev/null 2>&1 &
            fi
# waiting for start completely
```

Note that when binding to NUMA, it is necessary to bind the full path of the process. For example, if binding to a gbased process, configure its full path/opt/<IP>/node/server/bin/gbased to the corresponding/opt/<IP>/node/server/bin/gcluster_ In the services file, replace<IP>with the real IP address.

Revise the second part as follows:

```
Add Code Red around 500 lines:
            # start data service
            if [ $node_type == 2 -o $node_type == 3 ]; then
                for ((count=0; count<${#DataServerName[@]}; count++))
                do
                    instance_ no=0
                            for inst in `echo $GBASE_ INSTANCES|sed
's/:/\n/g'|sort|uniq`
                            do
                                if [ $inst != "" ]; then
                                    #echo $inst
                                    . $inst
                                    declare -a DataServerBin
                                    DataServerBin[0]=$GBASE_
HOME/bin/gbased
                                    DataServerBin[1]=$GBASE_
HOME/bin/gc_ sync_ server
                                    #echo ${DataServerBin[count]}
                                    #__ start ${DataServerName[count]}
${DataServerBin[count]}
                                        __ start ${DataServerName[count]}
```

```
${DataServerBin[count]} ${instance_no}
                                        if [ "$?" != "$RET_SUCCESS" ]; then
                                            ret_ start=$RET_ START_ ALL_
ERROR
                                        fi
                                            ((instance_no++))
                                    fi
                                done
                        done
                Fi
```

Restart Cluster Service

```
$ cd /opt/GBase/172.16.3.61/gnode/server/bin
$ ./ gcluster_ services all start
$ numastat `pidof gbased`
Per-node process memory usage (in MBs)
PID                              Node 0          Node 1           Total
---------------  --------------- --------------- ---------------
25788 (gbased)                   139.56            8.03          147.59
27445 (gbased)                     8.37          135.30          143.67
---------------  --------------- --------------- ---------------
Total                            147.93          143.32          291.26
```

3.3.5 Multi instance service management

The syntax for starting and stopping all instances on a multi instance server:

$ gcluster_ services all start/stop

The specified service syntax for starting and stopping an instance:

gcluster_ Services Service Name_ IP start

```
Start the 65 node service:
$ gcluster_ services gbase_ 172.16.3.65 start
$ gcluster_ services syncserver_ 172.16.3.65 start
Start 62 node service
$ gcluster_ services gbase_ 172.16.3.62 start
$ gcluster_ services syncserver_ 172.16.3.62 start
```

3.4 Software uninstallation

 **be careful**

After uninstalling, all system data will be lost. Please backup the required data before uninstalling.

Operation scenario

This operation guides engineers to uninstall the cluster when this version is not needed.

prerequisite

- After uninstallation, all data in the cluster will be lost. Please backup the required data;
- Before uninstalling, please manually stop the relevant services of all nodes in the cluster;
- A user password is required to perform the uninstallation.

Operating Steps

Step 1

Use dba users to stop cluster services for all cluster nodes.

```
[ gbase@rhel73-1    ~]$ gcluster_ services all stop
Stopping gcrecover :                                       [    OK    ]
Stopping gcluster :                                        [    OK    ]
Stopping gbase :                                           [    OK    ]
Stopping gbase :                                           [    OK    ]
Stopping syncserver :                                      [    OK    ]
Stopping syncserver :                                      [    OK    ]


[ gbase@rhel73-1    ~]$ gcware_ services all stop
Stopping GCWareMonit success!
Stopping gcware :                                          [    OK    ]
```

Step 2

Enter the gcinstall directory and use the installation user to perform the uninstallation.

The uninstallation command syntax is as follows:

```
# ./ unInstall.py --silent=demo.options [passwordInputMode]
```

Table 317 Parameter Description

Parameter Name	Description
silent	Specify the cluster node information to be uninstalled;
passwordInputMode	Optional parameters, specifying the method of obtaining passwords, and implementing different methods of obtaining passwords through different parameters. If this parameter is specified, the password in demo. options does not need to be modified again. The value range is [file, pwdsame, pwddiff], and the default value is file: ● File: represents obtaining from a file or command line parameter, consistent with the original method, in which the password in the file is plaintext; ● Pwdsame: This parameter is used when the password is entered by the user from the terminal and the passwords of all nodes are consistent. For different user passwords, it is only entered once and applies to all nodes using this password; ● Pwddiff: This parameter is used when the password is input by the user from the terminal and the passwords between nodes are inconsistent. For different user passwords, each node is input once, which is suitable for different nodes to use different passwords;

Example:

The parameter values in the demo.options file are consistent with the current running cluster information. If it is IPV6, nodeid can go to $GCWARE_ Obtain from the gcware. conf file under BASE/config: the nodeID under totem is gcwareHostNodeID, and the one under gcware is coordinateHostNodeID.

```
$./ unInstall.py --silent=demo.options
These GCluster nodes will be uninstalled.
CoordinateHost:
172.168.83.11       172.168.83.12       172.168.83.13
DataHost:
172.168.83.11       172.168.83.12       172.168.83.13       172.168.83.14
Are you sure to uninstall GCluster ([Y,y]/[N,n])? y
        172.168.83.11 unInstall 172.168.83.11 successfully.
        172.168.83.12 unInstall 172.168.83.12 successfully.
172.168.83.13     unInstall 172.168.83.13 successfully.
        172.168.83.14 unInstall 172.168.83.14 successfully.
```

3.5 Upgrade cluster

Operating principles:

1) Upgrading requires obtaining a new license file. Please apply and prepare the license file in advance. For the specific application process and operation methods, please refer to the section 3.2.1 Obtaining a License;

2) The system does not support direct upgrade from V8.6. X to V9.5.3. X. It needs to be upgraded from V8.6. X to V9.5.2. X first, and then from V9.5.2. X to V9.5.3. X;

3) Prohibit parallel execution of commands such as upgrade, installation, and uninstallation, including execution on different sessions or cluster nodes within the same cluster node;

4) During the upgrade process, all nodes of the cluster must be kept online;

5) Ensure that the cluster has completed initialization before upgrading;

6) Ensure that all cluster events are processed and the cluster has stopped external services before upgrading;

7) FEVENTLOG is not allowed during the upgrade process.

3.5.1 Upgrading V8.5.1.2 cluster to V9.5.2. X cluster

The V8.5.1.2 version cluster does not allow direct upgrade to V9.5.X.X version. You need to first upgrade to V8.6.X.X version before upgrading from V8.6.X.X version to V9.5.2. X.

3.5.2 Upgrading V8.6. X.X cluster to V9.5.2. X cluster

Currently, only upgrading from V8.6.X.X version to V9.5.2. X version is supported.

Upgrading from V8.6.X.X version to V9.5.3. X version requires upgrading from V8.6.X.X version to V9.5.2. X version before upgrading between V9.5.X.X versions to V9.5.3. X.

3.5.2.1 Preparation before upgrading

3.5.2.1.1 View existing cluster information

First, check the cluster status:

```
[gbase@8a ~]$ gcadmin
CLUSTER STATE:  ACTIVE
CLUSTER MODE:   NORMAL

============================================================================
|             GBASE COORDINATOR CLUSTER INFORMATION                        |
============================================================================
|   NodeName   |      IpAddress     |gcware |gcluster |DataState |
----------------------------------------------------------------------------
| coordinator1 |    192.168.0.1     | OPEN  |  OPEN   |    0     |
----------------------------------------------------------------------------
| coordinator2 |    192.168.0.2     | OPEN  |  OPEN   |    0     |
----------------------------------------------------------------------------
============================================================
|               GBASE DATA CLUSTER INFORMATION             |
============================================================
|NodeName |      IpAddress     |gnode |syncserver |DataState |
------------------------------------------------------------
|  node1  |    192.168.0.1     | OPEN |   OPEN    |    0     |
------------------------------------------------------------
|  node2  |    192.168.0.2     | OPEN |   OPEN    |    0     |
------------------------------------------------------------
|  node3  |    192.168.0.3     | OPEN |   OPEN    |    0     |
------------------------------------------------------------
|  node4  |    192.168.0.4     | OPEN |   OPEN    |    0     |
------------------------------------------------------------
```

 **be careful**

It is necessary to ensure that the status of all nodes is normal and there is no application access, otherwise the fault needs to be resolved and the business access needs to be stopped before continuing the operation.

Enter the database for operation:

 # su - gbase

$ gccli -uroot -p

Enter password:

GBase client 8.6.2.43 build 115142. Copyright (c) 2004-2019, GBase. All Rights Reserved.

```
//View existing cluster versions
gbase> SHOW VARIABLES LIKE '%VERSION';
+----------------------------+----------+
| Variable_ name             | Value    |
+----------------------------+----------+
| gbase_ kafka_ broker_ version |          |
| gbase_show_version         | 1        |
| gcluster_ hash_ version    | 1        |
| protocol_ version          | 10       |
| version                    | 8.6.2.43 |
+----------------------------+----------+
5 rows in set
```

3.5.2.1.2 Restore cluster data status

步骤 1

Use the gcadmin command to view the cluster status and determine if there is any unrecovered data or EVENT based on the DataState value. If the DataState value of all nodes is 0, you can skip the steps in this section.

```
[gbase@8a ~]$ gcadmin
CLUSTER STATE:  ACTIVE
CLUSTER MODE:   NORMAL

=============================================================
|             GBASE COORDINATOR CLUSTER INFORMATION         |
=============================================================
|   NodeName   |      IpAddress     |gcware |gcluster |DataState |
-------------------------------------------------------------
| coordinator1 |    192.168.0.1     | OPEN  |  OPEN   |    0     |
-------------------------------------------------------------
| coordinator2 |    192.168.0.2     | OPEN  |  OPEN   |    0     |
-------------------------------------------------------------
=============================================================
|               GBASE DATA CLUSTER INFORMATION              |
=============================================================
|NodeName |      IpAddress     |gnode |syncserver |DataState |
-------------------------------------------------------------
|  node1  |    192.168.0.1     | OPEN |   OPEN    |    0     |
-------------------------------------------------------------
|  node2  |    192.168.0.2     | OPEN |   OPEN    |    0     |
-------------------------------------------------------------
|  node3  |    192.168.0.3     | OPEN |   OPEN    |    0     |
-------------------------------------------------------------
|  node4  |    192.168.0.4     | OPEN |   OPEN    |    0     |
-------------------------------------------------------------
```

步骤 2

If there are nodes with a DataState value other than 0, the data of these nodes needs to be

restored. Firstly, you need to wait for automatic recovery. If automatic recovery is unsuccessful, you will need to manually perform data or EVENT recovery.

步骤 3

After completing log recovery, it is necessary to check whether the FEVENTLOG of all nodes has been successfully restored.

```
[gbase@8a gcinstall]$ ./gcinstall.py --silent=demo.options -U
********************************************************************************
Thank you for choosing GBase product!
Please read carefully the following licencing agreement before installing GBase product:
TIANJIN GENERAL DATA TECHNOLOGY CO., LTD. LICENSE AGREEMENT

    ... ... ...

********************************************************************************
********************************************************************************
                    Welcome to install GBase products
********************************************************************************
Error: gcinstall.py(line 1813) -- /var/lib/gcware/REDOLOG.* is not empty,please execute the order 'service
gcware restart'
```

3.5.2.1.3 Stop Cluster Service

Before performing the cluster upgrade operation, all node cluster services need to be stopped first.

The specific operation is as follows (each other node needs to perform this operation once):

$ gcluster_ services all stop

Stopping gcrecover : [OK]

Stopping gcluster : [OK]

Stopping gbase : [OK]

Stopping gbase : [OK]

Stopping syncserver : [OK]

Stopping syncserver : [OK]

[gbase@rhel73-1 ~]$ gcware_ services all stop

Stopping GCWareMonit success!

Stopping gcware : [OK]

3.5.2.2 Upgrade cluster

The specific upgrade steps are as follows:

步骤 1

Confirm that the database services of all cluster nodes have stopped.

步骤 2

Unzip the V9.5.X.X cluster installation package and switch to the gcinstall directory where the installation package has been unzipped.

步骤 3

Use root user to execute environment deployment scripts for gbase users on various nodes of the cluster:

```
# cd gcinstall
#Scp SetSysEnv. py gbase @ Cluster Node IP:/opt/
# ./ SetSysEnv.py --installPrefix=/opt --dbaUser=gbase
```

Please refer to the section 3.2.2 Initial Installation for SetSysEnv syntax and parameter descriptions.

Find and modify the demo. options configuration file in the gcinstall directory. The parameter values in the demo.options file are consistent with the original cluster information. GcwareHost must fill in the gcware node IP of the original cluster (in the 8.6.2. X cluster, the gcware and coordinator nodes are the same node), and the gcwareHostNodeID can be left blank under IP4. If it is IPV6, the nodeID can go to $GCWARE_ Obtained from the gcware. conf file under BASE/config: the nodeID under totem is gcwareHostNodeID, and the one under gcware is coordinateHostNodeID.

```
#su - gbase
$ vi /opt/gcinstall/demo.options
installPrefix= /opt
coordinateHost = 192.168.146.20,192.168.146.21,192.168.146.22
coordinateHostNodeID = 20,21,22
dataHost =
192.168.146.20,192.168.146.21,192.168.146.22,192.168.146.23,192.168.146.40,1
92.168.146.41,192.168.146.42,192.168.146.43
#existCoordinateHost =
#existDataHost =
#existGcwareHost=
gcwareHost = 192.168.146.20,192.168.146.21,192.168.146.22
gcwareHostNodeID = 20,21,22
dbaUser = gbase
```

```
dbaGroup = gbase
dbaPwd = 'gbase'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
#characterSet = utf8
#dbPort = 5258
#sshPort = 22
```

步骤 4

Execute using dbaUser user

./gcinstall.py --license_ File=gbase. lic -- silent=demo. options - U for cluster upgrade. If the upgrade is successful, the cluster will automatically start, and if the upgrade fails, the cluster will automatically fall back to version V8.6.X.X.

步骤 5

Use the root user to execute the GBase user's environment deployment script again on each node of the cluster:

```
# cd gcinstall
#Scp SetSysEnv. py gbase @ Cluster Node IP:/opt/
# ./ SetSysEnv.py --installPrefix=/opt --dbaUser=gbase
```

Please refer to the section 3.2.2 Initial Installation for SetSysEnv syntax and parameter descriptions.

3.5.2.3 Precautions after upgrading

- Upgrading the cluster from a version without uid (before 8.6.2.10) to a new version, the owner of the table is missing, meaning that the uid fields are all 0. Users need to manually execute the alter table<table_ name> set owner <user_ name>; Modify the owner of the table. If not modified, it will affect the use of resource management disk space control functions.
- If upgrading to 9.5.3 and requiring the use of multiple instances, please execute SetSysEnv.py again after successfully executing the upgrade command to configure the environment variables: Python SetSysEnv.py -- installPrefix=/opt -- dbaUser=gbase. If not executed, it will affect the use of log archiving function under multiple instances.

3.5.3 Version upgrade between V9.5.X.X clusters

When upgrading versions between V9.5.X.X clusters, it is necessary to set coordinatiteHost and dataHost to all existing coordinator and data cluster node IPs. If upgrading to 9.5.3. X, you also need to set gcwareHost to an existing gcware node. The upgrade operation needs to be performed on an existing Coordinator node.

Before proceeding with the version upgrade operation, it is necessary to ensure that all nodes are in a normal state and have completed cluster initialization. Otherwise, the fault needs to be resolved before proceeding with the operation.

Check the cluster information status and restore the FEVENT LOG step, which is the same as upgrading to V9.5.X.X version from V8.6.X.X version.

The specific steps are as follows:

步骤 1

Use DBAUser user to stop all cluster node services on all nodes in the cluster.

```
[ gbase@rhel73-1    ~]$ gcluster_ services all stop
Stopping gcrecover :                                       [    OK    ]
Stopping gcluster :                                        [    OK    ]
Stopping gbase :                                           [    OK    ]
Stopping gbase :                                           [    OK    ]
Stopping syncserver :                                      [    OK    ]
Stopping syncserver :                                      [    OK    ]

[ gbase@rhel73-1    ~]$ gcware_ services all stop
Stopping GCWareMonit success!
Stopping gcware :                                          [    OK    ]
```

步骤 2

Unzip the V9.5.X.X cluster installation package and switch to the gcinstall directory.

```
$cd /opt
$tar xjf    GBase8a_ MPP_ Cluster-License-9.5.3.27-redhat7.3-x86_ 64.tar.bz2
$cd /opt/gcinstall
```

步骤 3

Modify the demo. options configuration file.

```
#su - gbase
$ vi /opt/gcinstall/demo.options
installPrefix= /opt
coordinateHost = 192.168.146.20,192.168.146.21,192.168.146.22
coordinateHostNodeID = 20,21,22
dataHost =
192.168.146.20,192.168.146.21,192.168.146.22,192.168.146.23,192.168.146.40,1
92.168.146.41,192.168.146.42,192.168.146.43
#existCoordinateHost =
#existDataHost =
#existGcwareHost=
gcwareHost = 192.168.146.20,192.168.146.21,192.168.146.22
gcwareHostNodeID = 20,21,22
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbase'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
#characterSet = utf8
#dbPort = 5258
#sshPort = 22
```

be careful

- Require configuration information to be consistent with the original cluster, including character set settings;
- The IP addresses of the CoordinaterHost and DataHost nodes must be consistent with those before the upgrade;
- If upgrading to 9.5.3. X, you also need to set gcwareHost to an existing gcware node.
- If upgrading to 9.5.3. X and using IPV6, pay attention to obtaining the gcwareHostNodeID, which can be found in $GCWARE_ Obtained from the gcware. conf file under BASE/config: the nodeID under totem is gcwareHostNodeID, and the one under gcware is coordinateHostNodeID.
- During the upgrade, the cluster will automatically back up necessary files. The default backup directory is/home/$dbaUser. If necessary, you can use the parameter - backup_ Dir specified.
- Apply for the license file in advance.

步骤 4

Execute using dbaUser user/ gcinstall.py --license_ File=gbase. lic -- silent=demo. options - U for cluster upgrade. If the upgrade is successful, the cluster will automatically start. If the upgrade fails, the cluster will automatically fall back to the old version and can operate normally.

```
gcinstall.py [options]
Options:
-U. Upgrade
--Silent=SILENTCONFIG Installation Configuration File
    --license_ file=LICENSE_ FILE License file
    --backup_ dir=BACKUP_ The path for DIR automatic backup, default
to/home/$dbaUser
    -s, --skip_ Audit upgrade does not back up audit.log
```

3.6 Fallback cluster

3.6.1 Manual rollback from V9.5.X.X version cluster to pre upgrade version cluster

 **warning**

We do not recommend rolling back the cluster from version V9.5.X.X to the pre upgrade version, and in most scenarios, this rollback is invalid and can lead to data errors.

If the following conditions are met, the V9.5.X.X version cluster can be successfully rolled back to the pre upgrade version cluster:

1. The V9.5.2. X installation package decompression directory used when upgrading from the V8.6. X. X cluster to V9.5.2. X still exists. When upgrading from V8.6.X.X to V9.5.2. X, the system user data files of the previous version will be temporarily retained in the home directory of DBAUser.

2. After upgrading to the V9.5.X.X cluster, there were no DDL operations, expansion operations, generation of new distribution operations, or generation of new FEVENTLOGs.

3. When upgrading to the V9.5.X.X cluster, the backup files saved by the upgrade tool still exist. When upgrading to a V9.5.X.X cluster, the backup file name contains the

gcluster keyword, such as gcluster_ backup_ 9.5.3.17.114764_ 20191110162202.tar.bz2。

If upgrading to 9.5.3. X, there will also be independent backup files for gcware.

Firstly, all node services in the cluster must be stopped, and then the command to rollback the cluster version must be executed under the gbase user, as shown below:

```
python Restore.py --backupFile=/home/gbase/gcluster_ backup_
9.5.3.17.114764_ 20191110162202.tar.bz2 --silent=demo.options
[--passwordInputMode=PASSWORDINPUTMODE]
```

Table -318 Parameter Description

Parameter Name	Description
--backupFile	Specify the cluster backup files (gcluster and gnode files) to be rolled back. The default is under/home/$dbaUser.
--silent	Specify Configuration File
passwordInputMode	Optional parameters, specifying the method of obtaining passwords, and implementing different methods of obtaining passwords through different parameters. If this parameter is specified, the password in demo. options does not need to be modified again. The value range is [file, pwdsame, pwddiff], and the default value is file: ● File: represents obtaining from a file, in which the password in the file is plaintext; ● Pwdsame: This parameter is used when the password is entered by the user from the terminal and the passwords of all nodes are consistent. For different user passwords, only one input is required; ● Pwddiff: This parameter is used when the password is entered by the user from the terminal and the passwords between nodes are inconsistent. For different user passwords, each node is input once.

4 Administrator's Guide

This chapter introduces how to manage clusters. For system administrators to perform daily cluster health checks, cluster management, security management, audit management, backup and recovery operations.

4.1 Introduction to Component Tools

GBase 8a MPP Cluster provides management and analysis functions for massive data, as well as tools for easy monitoring of database clusters.

- Graphical tools for database administrator to monitor database clusters:

  Unified data platform monitoring and operation maintenance system

- Tools for developers to access, control, and manage database objects:

  GBaseDataStudio graphical management tool

  Gccli command line connection database tool

- Command line tools for database administrator to manage database clusters:

  Gcadmin cluster management tool

  gcluster_ Services cluster startup and shutdown tool

  gcware_ Services gcware start stop tool

- Tools for database administrator to backup and restore cluster data:

  Gcrcman backup and recovery tool

- It is convenient for database administrator to allocate system resources:

  Resource management function

- Tools for database administrator and developer data migration:

  Gcdump database object structure export tool

  SQL statement cluster data import and export function

  Orato8a Data Extraction Tool

  Db2to8a data extraction tool

  RTSync heterogeneous database full and incremental data synchronization tool

4.1.1 Introduction to Unified Data Platform Monitoring and Operation and Maintenance System

Function Introduction

The unified data platform monitoring and operation system supports monitoring of single or multiple clusters. Based on the alarm strategy set by the user, information such as system resource utilization, network communication, process operation, and cluster operation status of cluster nodes can be collected and monitored. The alarm information can be pushed to the user, enabling them to discover and troubleshoot cluster faults in a timely manner. The unified data platform monitoring and operation and maintenance system will also collect information persistence into the database, and conduct

multi-dimensional analysis and display of the cluster performance, so that users can tune or troubleshoot the cluster.

The function introduction is as follows:

- View the overall status: cluster mode, locks, alarm information, node status, session statistics, disk space usage, and loading information.
- Cluster server topology display: including node status and alarm level.
- Detailed data display of individual server monitoring indicators.
- Cluster session information query: includes searching for session status of specified nodes, removing sessions, SQL execution plans, etc.
- View SQL logs: Historical log information during cluster operation.
- System log collection: Various types of log information generated during cluster operation, with configurable log types.
- Cluster process control: can start/stop specified processes, and processes can be configured.
- Alarm information management: Regularly obtain alarm information, and users can view historical alarm information.
- Cluster database monitoring: Database list, table information of the specified database, column, index, and data distribution corresponding to the database and table, all of which can be queried through conditions.
- Resource statistics: including the performance of all nodes' disks, networks, CPUs, etc. in the cluster, the amount of cluster data, and the number of tasks executed by DDL, DML, DQL, etc.

Tool deployment

The unified monitoring tool mainly includes three functional modules: collection agent, collection center, and monitoring website.

- The collection agent module contains the GAgent component and needs to be deployed on all nodes of the cluster. This module is responsible for collecting the operating system, disk, memory, CPU, network traffic, node operation status, node processes, and cluster operation status of cluster nodes.
- The collection center module contains the GCenter component, which is responsible for persistence the information collected by the collection agent module to the resource library and processing node alarms. A collection center can only collect agents under the same cluster, and a collection center can monitor up to 100 collection agents. When the cluster size is large, multiple collection centers can be deployed to improve monitoring performance. This component is deployed on a Linux server.
- The monitoring website module contains the gcmonitor component and needs to be deployed on a Linux server. The gcmonitor component implements management functions for monitoring the entire cluster, displaying the overall running status and

performance of the cluster, data distribution of the cluster, cluster node alarms, and platform management functions for unified monitoring.

Preparation for deploying a unified monitoring website:

- Request to prepare a Linux server for installing the monitoring website. Require the server system username and password to be consistent with the cluster node.
- It is required to prepare one or more Linux servers for installing the collection center. If the cluster size is small, a collection center can be installed, and at this time, the collection center and monitoring website can also be installed on the same server.
- Require network interoperability between monitoring websites, collection centers, and cluster node servers.
- Request to prepare a server with an installed resource library. It is recommended to use the GBase 8a database as the resource library.
- Require all servers to have their SSH services turned on normally.
- The machine system time for installing monitoring websites, collection centers, and resource libraries is required to be synchronized with the system time of each node in the cluster.

explain

Unified monitoring provides automatic installation scripts for autoInstall to install all components of the collection agent, collection center, and monitoring website. At the same time, it also needs to rely on third-party components. Monitoring websites requires reliance on jre, tomcat, sysstat components, and resource library components.

Login tool

Step 1: Start the relevant services of the collection agent, collection center, and monitoring website and verify their correctness.

Step 2 confirms that the resource library has been properly initialized.

Step 3 Website running environment: Currently supports Firefox browser, Google Chrome browser, and IE browser. IE browser recommended version 10 or higher. We recommend using the latest version of Firefox browser.

Step 4: Enter the unified monitoring website address in the browser to log in to the monitoring website.

Http://[ip]: [port]/gcmonitor (such as http://192.168.5.174:8080/gcmonitor)

You need to change the IP address and port number to the IP address and port number of the tomcat server on which you installed the website.

Figure -41 Unified Monitoring User Login Interface

After the successful installation of **step 5** unified monitoring, the default initialized username is admin and the default password is 111111.

The admin user is a super user who is bound to the roles of "platform administrator" and "cluster supervisor" by default, and has the highest management authority for unified monitoring.

Step 6: Deploy the basic configuration of monitoring for the first time.

When you first deploy unified monitoring and want to monitor a GBase 8a MPP cluster, it is recommended that you follow the following steps to quickly create a platform to manage basic data and achieve cluster monitoring. The operation steps are:

Create cluster ->Create collection center ->Add server ->Create/modify user,

You can directly view cluster monitoring information by logging in to the monitoring website in the future.

reference resources

For the specific installation, deployment, and monitoring operations of the unified data platform monitoring and operation system, please refer to the
"User Manual for the Unified Data Platform Monitoring and Operation System".

4.1.2 Introduction to GBaseDataStudio Management Tools

Function Introduction

GBaseDataStudio management tool is a cross platform (suitable for Windows, Linux,

etc.) management tool provided by GBase 8a MPP. It combines a diverse set of graphical tools with multiple fully functional SQL script editors for accessing, controlling, and managing GBase 8a MPP Cluster.

The GBaseDataStudio management tool can complete the following tasks:

- Manage a single cluster environment or multiple cluster environments.
- Manage single or multiple cluster node servers in a single cluster environment.
- By registering cluster nodes, dynamic expansion of cluster management can be achieved.
- Visualize the management of objects such as databases, tables, indexes, views, stored procedures, and functions in a cluster environment.
- Visualize the management of cluster environments by creating, editing, and deleting users.
- Visualize the logs of the cluster environment.
- Manage data records in the cluster environment data object table.

Tool deployment

The GBaseDataStudio management tool is portable application, which can be used after direct decompression.

Login tool

Step 1: To open the GBaseData Studio management tool for the first time, you need to create a new connection before using it.

Using the "Connection Management" function, you can create, modify, and delete database connections.

Step 2: Perform relevant operations on the connected cluster database.

reference resources

For an introduction to the installation and use of the GBase Data Studio management tool, please refer to the GBase 8a MPP Cluster Management Tool Manual.

4.1.3 Introduction to gccli command-line tool

Function Introduction

Gccli is a command-line tool that comes with GBase 8a MPP Cluster and can also be deployed independently. This tool enables the execution of SQL statements and external SQL files.

reference resources

For detailed parameters and usage of the gccli tool, please refer to Chapter 4.4.1 of this manual, gccli.

4.1.4 Gcrcman backup and recovery tool

Function Introduction

GBase 8a MPP Cluster provides a dedicated backup and recovery tool (gcrcman) that supports full backup, incremental backup, full recovery, and recovery to specified backup points at the instance level, library level, and table level. It also supports displaying backup information, allowing users to easily backup and recover data throughout the entire cluster. Gcrcman is automatically installed with the installation of the cluster, on $GCLUSTER_ In the HOME/bin directory.

- Support full backup at cluster level, database level, and table level
- Support incremental backup at cluster level, database level, and table level
- Support cluster level, database level, and table level recovery to specified backup points for specified backup cycles
- Support cluster level, database level, and table level recovery to the latest backup cycle and backup point
- Support for remote backup and recovery (NFS mounts remote backup and recovery data to disk)
- Support for deleting backups and clearing invalid backup data
- Support for viewing backup information
- Support for deleting backup data
- Support for deleting junk backup data

reference resources

For detailed parameters and usage of gcrcman, please refer to Chapter 4.8 of this manual for backup and recovery management.

4.1.5 Introduction to data migration Tools

GBase 8a MPP Cluster provides a variety of user-friendly data migration tools according to the needs of different scenarios.

4.1.5.1 Database Object Structure Export

Function Introduction

The gcdump tool can export the structure of database objects:

- Export Table Structure
- Export Stored Procedure
- Export Custom Functions

explain

The gcdump tool is located at $GCLUSTER_ Under the HOME/bin path. By parameter gbase_ show_ ident_ case_ Sensitive can control the casing of exported column names, which is consistent with the casing of column names in the source table structure by default. Please refer to the configuration parameters of Gnode in Chapter 7.6.3 for details.

grammar

```
gcdump [OPTIONS] database [tables]
gcdump [OPTIONS] --databases [OPTIONS] DB1 [DB2 DB3...]
gcdump [OPTIONS] --all-databases [OPTIONS]
```

Table 41 Parameter Description

Parameter Name	explain
-u	Login cluster username
-p	Login cluster password
-R	Output stored procedures and functions
-B	Can output multiple databases
-W	Specify VC name

Example

```
$ $GCLUSTER_ BASE/server/bin/gcdump –uroot –p****** –B –R ssbm>/home/gbase/ssbm.sql
```

4.1.5.2 Data loading

Function Introduction

GBase 8a MPP Cluster provides a user oriented SQL interface loading method.

Supports the following functions:

- Support local file loading

- Support pulling and loading data from a universal data server;
- Support multiple protocols such as FTP/HTTP/HDFS/SFTP;
- Support parallel loading of multiple loaders on a single table to maximize loading performance;
- Supports loading of data files in various formats such as regular text, gzip compression, snappy compression, lzo compression, etc;
- Supports loading of regular text and fixed length text, and is compatible with V8.5.1.2 and V86 formats;
- Support real-time query of loading status and information
- Support error data traceability function, which can accurately locate the location of error data in the source file;
- The loading performance can continue to improve with the expansion of the cluster size.

grammar

```
LOAD DATA INFILE 'file_ list' INTO TABLE[vcname.] [dbname.]tbl_ name [options]
```

 **explain**

file_ List Description

- Cluster local data source loading:

  1) Supports specifying local files on one or more data nodes for loading. use file://host +abs_ Path, multiple file://host +abs_ The paths are separated by commas and support direct read mode to load local files of specified cluster data nodes.

  2) Support the simultaneous loading of files on each node by specifying all data nodes. Using file://+abs_ Path, multiple files://+abs_ The paths are separated by commas and support direct read mode to load local files of all data nodes in the cluster.

- Loading data sources on general file servers outside of the cluster:

  1) The FTP/HTTP/hdfs/sftp service needs to be built on the universal file server, and the data files need to be copied to the path configured by the service to ensure that cluster nodes can access the data through the corresponding service.

  2) When loading, use a URL to specify the data file path on a universal file server, using commas (',') as the separator for multiple files/directories, with the format " scheme://host:port/path , scheme://host:port/path "

  At the same time, both the file name and directory sections support the use of wildcards,

which default to matching paths and files, such as:“ http://10.10.1.1/data/??????/*.tbl” 。

Example

Example:

```
LOAD DATA INFILE ' ftp://gbase:gbase @127.0.0.1/data/a.tbl' INTO TABLE test.t DATA_ FORMAT 3;
LOAD DATA INFILE ' http://127.0.0.1/data/b.tbl.gz ' INTO TABLE test.t DATA_ FORMAT 3;
LOAD DATA INFILE ' hdp://gbase @127.0.0.1:50070/data/a.tbl.snappy' INTO TABLE test.t DATA_ FORMAT 3;
LOAD DATA INFILE ' ftp://192.168.0.1/pub/lineitem.tbl ， http://192.168.0.2/lineitem.tbl ' INTO TABLE test.lineitem FIELDS TERMINATED BY '|' ENCLOSED BY '"' LINES TERMINATED BY '\n';
```

Example of loading status and result viewing:

```
--Load status monitoring
gbase> use information_ schema;
gbase> select * from load_ status;
--Show tasks_ The following 5 error data messages for task id 100 starting from item 1
gbase> show load logs 100 limit 1,5;
--Display tasks on all coordinator nodes_ Top 10 error data messages for task ID 101
gbase> show gcluster load logs 101;
--Query all coordinator nodes, select query format, query loading information, and table name: CUSTER_ LOAD_ RESULT
gbase> select * from information_ schema.cluster_ load_ result;
```

4.1.5.3 Data export

Function Introduction

GBase 8a MPP Cluster provides data export function, using SELECT INTO OUTFILE ... Export in SQL syntax format, supporting:

- Export the data to the server side of the cluster, the designated FTP/SFTP server, or the Hadoop cluster, which can be exported as a text file or compressed file in gz/snappy/lzo format;
- Provide remote data export function, that is, export data from the cluster server to the machine where the cluster client is located, and the exported data is a text file.

grammar

```
SELECT ... INTO OUTFILE 'file_ name' [OPTION] FROM ...;
SELECT... FROM... INTO OUTFILE 'file_ name' [OPTION];
rmt:SELECT... FROM... INTO OUTFILE 'file_ path' [OUTFILE_OPTION];
```

Example

```
gbase> select * from aa into outfile '/home/davies/out.txt' fields escaped by '' terminated by '|'
double_ enclosed by '"' null_value 'null';
```

reference resources

For detailed usage of loading and exporting, please refer to Chapter 5.2 Data Integration and Data Management of this manual.

4.1.5.4 Data migration

4.1.5.4.1 Introduction to Orato8a Data Extraction Tool

Introduction to Orato8a

Orato8a is a special tool that can quickly and efficiently extract data from the oracle database system and save the data to a specified file or directly migrate it to the GBase 8a MPP Cluster. Orato8a also provides two modes of query statement export and full table export. The login user of full table export needs to access the dba in the oracle database_ extents、dba_ Objects and dba_ These three tables have select permission.

Orato8a deployment

Orato8a is a standalone data extraction tool that needs to be deployed on a machine that can access Oracle (that is, it needs to be deployed with the Oracle client), or directly on a server with the Oracle server.

The Orato8a installation package is provided in tar. bz2 compressed format. For example: orato8a_ 26794_ Redhat6.2_ x86_ 64.tar.bz2。

After decompression, an orato8a executable program file will be generated in the decompression directory.

```
# tar xfj orato8a_ 26794_ Redhat6.2_ x86_ 64.tar.bz2
$ ll
Total usage 2068
……
-Rw-r - r -1 root 1380535 August 23 01:08 orato8a
```

Orato8a syntax

```
./orato8a parameter_ 1 parameter_ 2 …… parameter_ n
```

 **explain**

The user executing Orato8a must be a user with access to the oracle database. When exporting blob or clob type data columns in Oracle, different versions of Orato8a have different parameter controls, which need to be handled according to the specific version reference manual.

Orato8a Example

```
$ ./ orato8a --user='ct1/ ct1ct1@orcl ' --query="select LO_ORDERKEY, LO_LINENUMBER
FROM lineorder_test" --file='/opt/orato8a_ output/lineorder.txt ' --field=";" --format=3
    export columns: 2
    export rows: 10
    export time: 0 sec
    process ok!
```

4.1.5.4.2 Introduction to db2to8a Data Extraction Tool

Introduction to db2to8a

Db2to8a is a specialized tool that can quickly and efficiently extract data from a database system, and the extracted data can be saved to a specified file.

Db2to8a deployment

Db2to8a is a standalone data extraction tool that needs to be deployed on a client machine that can access db2and directly on the same server as the db2server.

The db2to8a installation package is provided in tar. bz2 compressed format. For example: db2to8a_ 24816_ Redhat6.2_ x86_ 64.tar.bz2。
After decompression, an executable program named db2to8a will be directly generated in the decompression directory.

```
# tar xfj db2to8a_ 24816_ Redhat6.2_ x86_ 64.tar.bz2
$ ll
```

```
Total usage 2068
……


-Rw-r - r -1 root 1380535 August 23 01:08 db2to8a
-Rw-r - r -1 root 663929 August 22nd 17:13
```

Db2to8a syntax

```
./db2to8a parameter_1 parameter_2 …… parameter_n
```

explain

The user executing db2to8a must be a user who can access the database.

Example of db2to8a

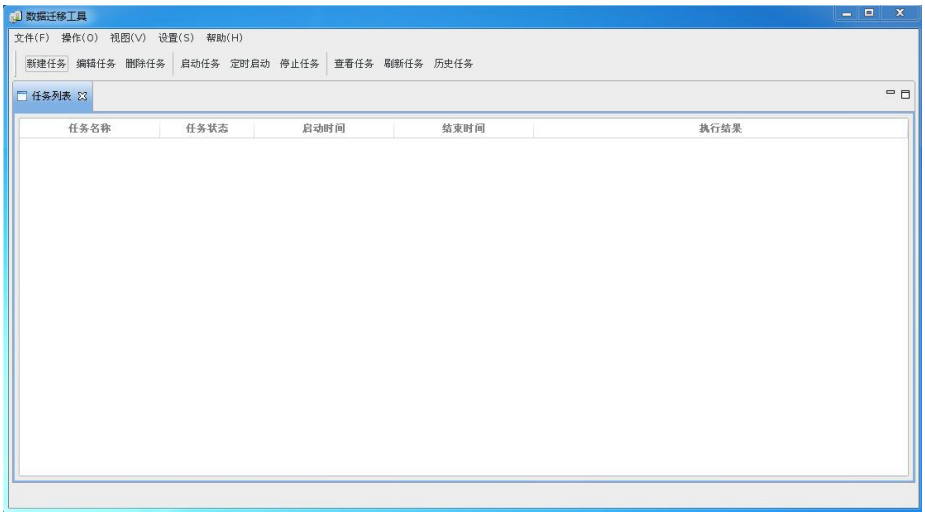
```
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select *    from t" -f'data1.txt' -m'3' -e'|' -l'\n'
-s'h'
you machine is Little endian!
   Connecting to test...
   Connected to test.
--- unload [text file] mode ---
--- field="|" ---
            0 rows exported at 2013-08-30 13:33:29
            7 rows exported at 2013-08-30 13:33:29
output file data1.txt closed
 export:              7 rows.
 export:              5 columns.
 export time:      0.00 sec
   Disconnecting from test...
   Disconnected from test.
```

4.1.5.4.3 Introduction to GBase Migration Toolkit

GBase Migration Toolkit migration tool is a tool provided by GBase that can realize data migration of heterogeneous databases. At present, data migration in the source database (currently supported source databases include ACCESS, Oracle, SQL Server2005, DM, DB2, MySQL, ShenTong, GBase8sV8.3, GBase8t, GBase8s,

PostgreSQL, and Teradata) can be migrated to the target database (currently supported target databases include GBase8a, GBase8t, and GBase8sV8.7).

The migration tool is a C/S structured software that is easy to install and can be used by simply obtaining and decompressing the installation package. The migration tool has a simple and easy to operate graphical interface. It can create corresponding tasks according to the requirements of data migration, and set the migration tasks accordingly to achieve multithreading and concurrent data migration.



4.1.6 Inter cluster synchronization tool

4.1.6.1 Tool Introduction

The inter cluster synchronization tool is a binary synchronization tool based on mirror clusters. The tool installation package is as follows:

gcluster_ rsynctool-9.5.2.28-redhat7.3-x86_ 64.tar.bz2

The synchronization object is the data in the library, which is synchronized by parsing and comparing changes in binary files; Including incremental synchronization and full synchronization. This tool has the following functional points:

1.Support incremental and full data synchronization methods;

2.Support read back verification of dropped disk data;

3.Support simultaneous synchronization of primary and backup shards (does not guarantee the data security of the backup cluster's tables);

4.Support synchronizing the main shard first, and then synchronizing the backup shard after the main shard is successful;

5.Support the use of regular database users for inter cluster synchronization.

be careful

- The versions of the active and standby clusters must be consistent;
- The main and backup clusters are isomorphic (including the same number of cluster nodes, hash distribution, table structure, and table sharding distribution);
- At least one set of available shards in the database tables that need to be synchronized between the active and standby clusters, and the cluster status is normal;
- The backup cluster's tables cannot provide external services during the synchronization process;
- The system table is not synchronized, and the screen output is' $dbname. $tbname is system table which needs not to be synchronized; '. The same record is also present in the log, with a record level of' info '.

4.1.7 UDF

UDF (User Defined Function) is a user-defined function that users can add themselves. Users can define and develop efficient SQL functions themselves through the universal extension mechanisms of UDF and UDAF (implemented using C/C++or Python languages).

reference resources

For detailed usage of UDF, please refer to Chapter 5.5.1 UDF&UDAF of this manual.

4.1.8 Transparent data access tool between clusters

The GBase 8a MPP Cluster achieves transparent data access between local GBase 8a MPP Cluster clusters and remote clusters (homogeneous data sources (i.e. GBase 8a MPP Cluster clusters outside the current cluster) or heterogeneous clusters (such as Oracle) through the combination of transparent gateways and DBlinks.

- The main function of the GBase 8a MPP Cluster Transparent Gateway is to connect to other clusters outside the GBase 8a MPP Cluster. After obtaining the request information of the db link, the data is extracted into the GBase 8a MPP Cluster or pushed to other clusters according to the request;
- The GBase 8a MPP Cluster DBLink tool enables remote cluster data to be associated with local cluster data through collaboration with transparent gateway services.

reference resources

For detailed usage of inter cluster data transparent access tools, please refer to 4.13 Inter cluster Transparent Gateway Tools and 4.14 DBLink Tools in this manual.

4.2 Service Management

4.2.1 Cluster initial user and login

Operation scenario

After the cluster is installed and running normally, you can proceed with the operations in this chapter.

prerequisite

List of operating system users and database users that exist in the system after installation.

Table 42 List of Existing Users in the System

User category	user name	Default password
Operating System Users	gbase	Manually specified
Database super account	root	empty

Operating Steps

Step 1

System users switch to GBase users.

```
# su - gbase
```

Step 2

Log in to the cluster and set the user login password.

By default, after the cluster installation is completed, the cluster will create two default database super accounts, root account and gbase account. After logging into GBase 8a MPP Cluster for the first time, the administrator must set a secure password for both the root and gbase accounts.

be careful

1、The root account can be deleted.

Example: Taking modifying the root account password as an example.

```
$    gccli –uroot
GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.    All Rights
Reserved.
gbase> SET PASSWORD FOR root = PASSWORD('H133%_h');
Query OK, 0 rows affected
#The command to log out is to type " q" at the gbase>prompt.
gbase> \q
Bye
#After changing the password of root, log in to the cluster again.
$    gccli -uroot -p
Enter password:
GBase client 9.5.2.17.111533. Copyright (c) 2004-2020, GBase.    All Rights
Reserved.
gbase >
```

4.2.2 Cluster Service Management

4.2.2.1 Cluster start stop tool

Operation scenario

The system administrator needs to check the status of database services or start or stop some database services.

Tool Introduction

- Tool name: gcluster_ services
  gcware_ services
- Tool storage path: $GCLUSTER_ BASE/server/bin/gcluster_ services
  $GCWARE_ BASE/sbin
- Function: Used to start and stop cluster related services

  gcluster_ Services start/stop gcluster and gnode related services

gcware_ Services start/stop gcware related services

- Command format:

```
Start and stop of gcluster and gnode services
gcluster_ services <gbase|gcluster|gcrecover|syncserver|all>  <start|stop
[--force]|restart [--force]|info>
gcluster_ services help
Gcware service startup and shutdown
gcware_ services <gcware|all>  <start|stop [--force]|restart [--force]|info>
gcware_ services help
```

Table -43 Parameter Description

Parameter Name	Description
gbase|gcluster|gcrecover|syncserver|all	Services included in all: Gcluster node services: gcluster, gcrecover Gnode node services: gbase, syncserver Monitoring services: gcmonit, gcmmonit
gcware|all	Services included in all: Gcware service: gcware Monitoring service: gcware_ monit、 gcware_ mmonit
start	Start corresponding service
stop	Stop corresponding services
restart	Restart the corresponding service
force	When the service cannot be stopped, the internal kill -9 method is used to forcibly stop the service process. Therefore, the - force option can only be used when the service cannot be stopped, and can only be used for stop and restart operations.

Tool usage

Example 1

Start and stop all services in the cluster.

- Enable all services:

```
$ gcluster_ services all start
Starting gbase :                                           [  OK  ]
Starting syncserver :                                      [  OK  ]
Starting gcluster :                                        [  OK  ]
Starting gcrecover :                                       [  OK  ]
$ gcware_ services all start
Starting gcware :                                          [  OK  ]
Starting GCWareMonit success!
```

- Stop all services:

```
gcluster_ services all stop
Stopping gcrecover :                                       [  OK  ]
Stopping gcluster :                                        [  OK  ]
Stopping gbase :                                           [  OK  ]
Stopping syncserver :                                      [  OK  ]
$ gcware_ services all stop
Stopping GCWareMonit success!
Stopping gcware :                                          [  OK  ]
```

- Restart all services:

```
$ gcluster_ services all restart
Stopping gcrecover :                                       [  OK  ]
Stopping gcluster :                                        [  OK  ]
Stopping gbase :                                           [  OK  ]
Stopping syncserver :                                      [  OK  ]
Starting gbase :                                           [  OK  ]
Starting syncserver :                                      [  OK  ]
Starting gcluster :                                        [  OK  ]
Starting gcrecover :                                       [  OK  ]
$ gcware_ services all restart
Stopping GCWareMonit success!
Stopping gcware :                                          [  OK  ]
Starting gcware :                                          [  OK  ]
Starting GCWareMonit success!
```

- To view the current execution status information of all services:

```
$ gcluster_ services all info
gcluster is running
gcrecover is running
gbase is running
syncserver is running
$ gcware_ services all info
gcware is running
```

Example 2

The gcluster service starts and stops.

- Start the gcluster service:

```
$ gcluster_ services gcluster start
Starting gcluster :                                        [  OK  ]
```

- Stop gcluster service:

```
$ gcluster_ services gcluster stop
Stopping gcluster :                                        [  OK  ]
```

- Restart the gcluster service:

```
$ gcluster_ services gcluster restart
Stopping gcluster :                                        [  OK  ]
Starting gcluster :                                        [  OK  ]
```

- To view the running status of the gcluster service:

```
$ gcluster_ services gcluster info
gcluster is running
```

Example 3

GBase service start and stop.

- Start the gbase service:

```
$ gcluster_ services gbase start
Starting gbase :                                           [  OK  ]
```

- Stop gbase service:

```
$ gcluster_ services gbase stop
Stopping gbase :                                           [  OK  ]
```

- Restart the gbase service:

```
$ gcluster_ services gbase restart
Stopping gbase :                                           [  OK  ]
Starting gbase :                                           [  OK  ]
```

- To view the running status of the gbase service:

```
$ gcluster_ services gbase info
gbase is running
```

Example 4

The gcware service starts and stops.

- Start gcware service command:

```
$ gcware_ services gcware start
Starting gcware :                                          [  OK  ]
```

- Stop gcware service command:

```
$ gcware_ services gcware stop
Stopping gcware :                                          [  OK  ]
```

- Restart gcware service command:

```
$ gcware_ services gcware restart
Stopping gcware:                                           [  OK  ]
Starting gcware:                                           [  OK  ]
```

- To view the running status of the gcware service:

```
$ gcware_ services gcware info
```

```
gcware is running
```

4.2.2.2 **Service monitoring tools**

introduce

The operation process of GBase 8a MPP Cluster requires starting services such as gbased, gclusterd, gcware, etc. in the system. These service programs may end their processes or be forcibly shut down by the system under certain special circumstances (such as system exceptions, excessive resource utilization, abnormal program operation, etc.). The cluster provides two monitoring tools to monitor and manage these service processes: gcmonit.sh and gcware_ monit.sh

- The main functions of gcmonit:

  1) Real time monitoring of gcluster and gnode service programs, mainly including gbased, gclustered, gcrecover, and gc_ sync_ Server). Once the process state of a service program is found to change, the corresponding command will be executed according to the content in the configuration file.

  2) Provide configuration files that can be modified by users. The configurable contents include: the name of the service program to be monitored or the startup command line of the process to be monitored, the method to be executed when the status of the service process state changes, the time interval to detect the service program, the path and name of the log file, etc.

  3) Record the start and stop information of each service.

  4) Implement high availability of gcluster and gnode services.

- The functions implemented by gcmmonit and gcmonit are completely identical, but their monitoring scope is different. gcmonit is responsible for monitoring the operation status of gcluster and gnode service programs and gcmmonit programs; And gcmmonit is only responsible for monitoring the running status of the gcmonit program.

- gcware_ The function of monit: it is mainly responsible for the real-time monitoring of gcware services. Once the gcware process state changes, it will execute the corresponding commands according to the contents of the configuration file.

- gcware_ Mmonit is mainly responsible for monitoring gcware_ Monit, achieving high availability of gcware services.

be careful

The configuration files for gcmonit and gcmmit are located in the following directory:

$GCLUSTER_ BASE/config/gcmonit.conf(gcmmonit.conf)

gcware_ Monit and gcware_ The configuration files for mmonit are located in the following directory:

$GCWARE_ BASE/config/gcware_ monit.conf (gcware_mmonit.conf)

If the configuration file is modified, the corresponding service needs to be restarted to take effect.

Log monitoring process logs

The gcmonit and gcmmonit log files default to $GCLUSTER_ Under BASE/log/gcluster, changes can be made through the configuration file.

gcware_ Monit and gcware_ The log file of mmonit defaults to $GCWARE_ Under BASE/log, changes can be made through the configuration file.

Command monitoring process related commands

```
gcmonit.sh <start | stop | restart | status [prog_name]>

gcware_ monit.sh <start | stop | restart | status [prog_name]>
```

Parameter Description:

prog_ Name: Indicates the program name that can be monitored.

- Start monitoring

```
$ gcmonit.sh start
Starting GCMonit success!
$ gcware_ monit.sh start
Starting GCWareMonit success!
```

- Turn off monitoring

```
$ gcmonit.sh stop
Stopping GCMonit success!
$ gcware_ monit.sh stop
Stopping GCWareMonit success!
```

- Restart monitoring

```
$ gcmonit.sh restart
Stopping GCMonit success!
Starting GCMonit success!
$ gcware_ monit.sh restart
```

```
Stopping GCWareMonit success!
Starting GCWareMonit success!
```

- Query GCMonit status

```
$ gcmonit.sh    status
+-------------------------------------------------------------------------+
|SEG_ NAME                    PROG_ NAME                    STATUS
PID
+-------------------------------------------------------------------------+
|gcluster                   gclusterd                   Running                9371
|gcrecover                   gcrecover                   Running                3917
|gcmmonit                     gcmmonit                     Running                4491
|gbase                        gbased                       Running                3940
|syncserver                  gc_ sync_ server            Running                4484
+-------------------------------------------------------------------------+

$ gcware_ monit.sh status
+-------------------------------------------------------------------------+
|SEG_ NAME                    PROG_ NAME                    STATUS
PID
+-------------------------------------------------------------------------+
|gcware                       gcware                       Running                31942
|gcware_ mmonit               gcware_ mmonit               Running                31800
+-------------------------------------------------------------------------+
```

4.2.3 Cluster Service Configuration

4.2.3.1 GCWare Basic Configuration

configuration file

On $GCWARE_ You can view and modify the basic configuration of GCWare in the BASE/config/gcware. conf configuration file.

Configuration File Format Description

```
totem {
    version: 2
    secauth: off
    interface {
        member {
            memberaddr: 192.168.146.20
        }
        member {
```

```
                memberaddr: 192.168.146.21
            }
            ringnumber: 0
            bindnetaddr: 192.168.146.20
            ttl: 1
        }
        transport: udpu
        leader_ heartbeat:200
        election_ timeout:2000
        server_ port:5918
        client_ port:5919
        max_ message_ size:1048576
        max_ redolog_ size:512
        data_ dir:/opt/192.168.146.20/gcware/data/gcware
        log_ dir:/opt/192.168.146.20/gcware/data/gcware
}
logging {
        fileline: off
        to_ stderr: no
        to_ file: yes
        to_ syslog: no
        logfile: /opt/192.168.146.20/gcware/log/gcware.log
        gcware_ system_ log: /opt/192.168.146.20/gcware/log/gcware_ system.log
        debug: off
        timestamp: on
        logger {
            ident: AMF
            debug: off
            tags: enter|leave|trace1|trace2|trace3|trace4|trace6
        }
}

gcware {
        persistent_ interval: 5
        check_ interval: 30
        whole_ check_ interval_ num: 20
        cfg_ connect_ timeout: 5000
        gcluster_ port: 5258
        gnode_ port: 5050
        syncserver_ port: 5288
        node_ ssh_ port: 10022
        check_ coordinator_ thread_ num: 1
        check_ dataserver_ thread_ num: 10
        enable_ node_ regist: 1
```

```
    enable_ check_ param: 0
    coordinator {
        member {
            memberaddr: 192.168.146.20
        }
        member {
            memberaddr: 192.168.146.21
        }
    }
}
```

Table -44 Parameter Description

Parameter Name	Description
【totem】	The configuration information of the distributed basic communication protocol totem protocol is related to the internal configuration of the gcware cluster
interface	Configure a set of IPs that need to monitor heartbeat, where memberaddr is the IP to be monitored, and here is the IP of each node in the gcware cluster. Multiple interfaces can be configured, and different interfaces are identified by ringnumbers to distinguish different heartbeats.
leader_ heartbeat election_ timeout	leader_ Heartbeat is the election heartbeat of the GCware cluster, with a default of 200ms; election_ Timeout is the timeout of the election heartbeat, with a default of 2000ms. The main node of the gcware cluster informs other nodes in the cluster that the main node is normal every 200ms, and the nodes in the gcware cluster undergo selection_ If no heartbeat is received from the master node within the timeout period of 2000ms, a new election will be initiated to elect a new GCware master node.
server_ port client_ port	GCware server port, default 5918 Gcware client port, default 5919 The port can be changed, and if changed, all gcware nodes need to be changed.
【logging】	Gcware log related information
【gcware】	Gcware detects configuration information related to the status and data consistency of each node in the 8a cluster (including gcluster and gnode clusters)
persistent_ interval	Internal consistency check of gcware cluster, default to 5s

Parameter Name	Description
whole_ check_ interval_ num check_ interval	Every check_ interval *Whole_ check_ interval_ Num (30s * 20 times, i.e. 600s) detects whether the service ports of all nodes are connected once to determine the status of each node's services Every check_ Interval (30s) detects whether abnormal nodes and abnormal services have returned to normal once
cfg_ connect_ timeout	The timeout time for detecting each node in the cluster is set to offline after the timeout
gcluster_ port gnode_ port syncserver_ port node_ ssh_ port	The corresponding ports used by gcware to detect whether various services of each node are functioning properly. If the port of the cluster service changes, it needs to be modified to the actual port used by the cluster service
check_ coordinator_ thread_ num check_ dataserver_ thread_ num	check_ coordinator_ thread_ Num is the number of concurrent threads detected by gcware on the gcluster node of the cluster, defaulting to 1 check_ dataserver_ thread_ Num is the number of concurrent threads detected by gcware when detecting gnode nodes in the cluster, with a default value of 10
enable_ node_ regist	GCware tracks the control parameters of the gnode mechanism through registration, which defaults to 0 and is not enabled. If this parameter is enabled, it is necessary to simultaneously enable the corresponding enable in the configuration files of each gnode node and gcluster node_ node_ The register parameter. For detailed information, please refer to 4.2.3.8 Gcware's configuration for monitoring gnode status through registration
enable_ check_ param	Gcware tracks the activation of the gnode mechanism through registration, and checks the enable status on each gnode node when the gnode service starts_ check_ Are the parameters specified by param consistent? If not, the gnode service cannot start normally. enable_ check_ The current supported values for param are: 'gbase_ segment_ size%gbse_ compression_ str_ method%gbase_ compression_ num_ method' Enable gcware_ node_ Enable after the register parameter is enabled_ check_ The param parameter is only valid

Example

Configuration file $GCWARE for GCWare_ The content reference for BASE/config/gcware. conf is as follows:

```
$ cat $GCWARE_ BASE/config/gcware.conf
totem {
        version: 2
        secauth: off
        interface {
                member {
                        memberaddr: 192.168.146.20
                }
                member {
                        memberaddr: 192.168.146.21
                }
                ringnumber: 0
                bindnetaddr: 192.168.146.20
                ttl: 1
        }
        transport: udpu
        leader_ heartbeat:200
        election_ timeout:2000
        server_ port:5918
        client_ port:5919
        max_ message_ size:1048576
        max_ redolog_ size:512
        data_ dir:/opt/192.168.146.20/gcware/data/gcware
        log_ dir:/opt/192.168.146.20/gcware/data/gcware
}
logging {
        fileline: off
        to_ stderr: no
        to_ file: yes
        to_ syslog: no
        logfile: /opt/192.168.146.20/gcware/log/gcware.log
        gcware_ system_ log: /opt/192.168.146.20/gcware/log/gcware_ system.log
        debug: off
        timestamp: on
        logger {
                ident: AMF
                debug: off
                tags: enter|leave|trace1|trace2|trace3|trace4|trace6
        }
}
```

```
gcware {
        persistent_ interval: 5
        check_ interval: 30
        whole_ check_ interval_ num: 20
        cfg_ connect_ timeout: 5000
        gcluster_ port: 5258
        gnode_ port: 5050
        syncserver_ port: 5288
        node_ ssh_ port: 10022
        check_ coordinator_ thread_ num: 1
        check_ dataserver_ thread_ num: 10
        enable_ node_ regist: 1
        enable_ check_ param: 0
        coordinator {
                member {
                        memberaddr: 192.168.146.20
                }
                member {
                        memberaddr: 192.168.146.21
                }
        }
}
```

4.2.3.2 GCluster Basic Configuration

configuration file

On $GCLUSTER_ BASE/config/gbase_ 8a_ In the gcluster.cnf configuration file, you can view and modify the basic configuration of GCluster.

On $GCLUSTER_ BASE/config/cluster_ The ID and other information of the node are recorded in common.cnf for viewing.

explain

- Unless otherwise specified, all configuration modifications to GCluster are made in GBase_ 8a_ In the [gbased] component within gcluster.cnf.

Configuration File Format Description

```
[TagName]
Variable_ name = Value
```

Table -45 Parameter Description

Parameter Name	Description

Parameter Name	Description
TagName	The component in effect for the variable to be set. ● Client: Client ● GBased: Related configurations of GCluster service ● GBasedump: Related configurations of GBasedump service ● GBase: Related configurations of GBase
Variable_ name	Variable name set
Value	Variable value set

Example

Configuration file $GCLUSTER for GCluster_ BASE/config/gbase_ 8a_ The content reference of gcluster.cnf is as follows:

```
$ cat $GCLUSTER_ BASE/config/gbase_ 8a_ gcluster.cnf
[client]
port=5258
socket = /opt/192.168.146.22/gcluster_ 5258.sock
connect_ timeout=43200
#default_ character_ set=gbk

[gbased]
basedir = /opt/192.168.146.22/gcluster/server
datadir = /opt/192.168.146.22/gcluster/userdata/gcluster
socket = /opt/192.168.146.22/gcluster_ 5258.sock
pid_ file = /opt/192.168.146.22/gcluster/log/gcluster/gclusterd.pid

#default_ character_ set=gbk

#gcluster_ metadata_ server_ ip=192.168.7.195
log_ error
port=5258
gcluster_ gnode_ port=5050
core_ file
default_ storage_ engine=express
default_ time_ zone='+8:00'
_ gbase_ query_ path=0

skip_ name_ resolve
query_ cache_ type = 0
query_ cache_ size = 0M
event_ scheduler= 1

thread_ stack = 4194304

sql_ mode=PAD_ CHAR_ TO_ FULL_ LENGTH,PIPES_ AS_ CONCAT,ANSI_
```

```
QUOTES,IGNORE_ SPACE,NO_ AUTO_ CREATE_ USER,NO_ AUTO_
VALUE_ ON_ ZERO,NO_ ENGINE_ SUBSTITUTION,STRICT_ ALL_
TABLES,NO_ ZERO_ IN_ DATE,NO_ ZERO_ DATE,ONLY_ FULL_ GROUP_
BY
lower_ case_ table_ names=1

max_ connections = 10000
max_ connect_ errors=1000000
max_ allowed_ packet = 64M
net_ write_ timeout = 1000000
net_ read_ timeout = 1000000
connect_ timeout = 1000000
interactive_ timeout    =1000000
wait_ timeout = 1000000
open_ files_ limit = 65535



gbase_ express_ log = 1

gcluster_ connect_ net_ read_ timeout    =    1000000
gcluster_ connect_ net_ write_ timeout =    1000000
gcluster_ connect_ timeout = 1000000
gcluster_ wait_ query_ cancel_ timeout = 200
gcluster_ reconn_ times = 3
gcluster_ async_ connect_ timeout = 120

gcluster_ use_ special_ insert_ method = 1
gcluster_ use_ special_ materialized_ table = 1
gcluster_ special_ insert_ method_ comment=temp

gcluster_ use_ new_ threadpool = 1
gcluster_ max_ thread_ in_ pool = 600

gcluster_ use_ conn_ pool = 1
gcluster_ max_ conn_ in_ pool = 300
gcluster_ conn_ ping_ expire = 0

gcluster_ dynamic_ cluster_ node_ status = 1
gcluster_ lock_ level = 2
gcluster_ temp_ table_ engine='express nolock'
gcluster_ dml_ ddl_ proxy_ switch = 0
gcluster_ adjust_ nodes_ before_ redist = 0

gcluster_ starschema_ optimize = 0
```

```
gcluster_ starschema_ join_ estimate_ optimize = 1
gcluster_ hash_ redistribute_ groupby_ optimize = 1
gcluster_ hash_ redistribute_ join_ optimize = 2
gcluster_ crossjoin_ use_ hash_ distribution = 1
gcluster_ insertselect_ use_ values_ optimize = 0
gcluster_ union_ optimize = 1
gcluster_ count_ optimize = 1
gcluster_ insert_ singlegrouppart_ optimize = 0

gcluster_ empty_ result_ set_ optimize = 0

gcluster_ special_ correlated_ optimize = 1
gcluster_ support_ hash_ redist_ combiner = 0
gcluster_ order_ by_ limit_ offset_ optimize = 0
gcluster_ mode_ wait = 1
gcluster_ mode_ checkinterval = 5
gcluster_ feventlog_ optimize = 1

gcluster_ ha_ event_ monitor = 1
gcluster_ ha_ node_ left_ event_ delay = 120000

gcluster_ sql_ statistics = 0
gcluster_ use_ new_ decimal = 1
gcluster_ query_ retry = 1
gcluster_ insert_ optimize_ flag = 1

gcluster_ serial_ exec_ query = 0
#gcluster_ special_ correlated_ optimize = 1
#gcluster_ ddl_ parallel_ execute =1

gbase_ compression_ str_ method=5
gbase_ compression_ num_ method=5

back_ log = 65535

gcluster_ hash_ version = 1

_ gbase_ transaction_ disable = 1

#kafka consumer parameters, turn on/off according to your requirement.
#gcluster_ lock_ level = 10
#gcluster_ assign_ kafka_ topic_ period=20
#gcluster_ kafka_ max_ message_ size=1000000000
#gcluster_ kafka_ batch_ commit_ dml_ count=100000
```

```
#gcluster_ kafka_ local_ queue_ size=210000
#gcluster_ kafka_ consume_ batch=10
#gcluster_ kafka_ parallel_ commit = 1
#gcluster_ kafka_ delete_ execute_ directly=0
#gcluster_ kafka_ loader_ max_ start_ count=20
#gcluster_ kafka_ user_ allowed_ max_ latency=3000
#gcluster_ kafka_ message_ format_ type=JSON
#gcluster_ kafka_ consumer_ enable=1
#gcluster_ kafka_ result_ check=1
#gcluster_ suffix_ consistency_ check=1
#gcluster_ kafka_ primarykey_ can_ be_ null=0
#_ t_ gcluster_ kafka_ null_ transform=0
#gcluster_ kafka_ consumer_ output_ charset_ name=UTF8

#const express can calculated before query
gcluster_ prepare_ const_ express = 0

[gbasedump]
max_ allowed_ packet = 64M



[gbase]
no_ auto_ rehash
```

Configuration file $GCLUSTER for GCluster_ BASE/config/ cluster_ The content reference of common.cnf is as follows:

```
{
	"cluster": {
		"uuid":"a7777256-7bf5-11eb-b80f-000c29b37bff",
		"nodeIPtype":"0",
		"localnodeid":"378710208",
		"nodes": [
		{
			"memberaddr":"192.168.146.22"
		},
		{
			"memberaddr":"192.168.146.23"
		}
		]
	},

	"datanode": {
		"nodes": [
		{
			"memberaddr":"192.168.146.22"
```

```
            },
            {
                "memberaddr":"192.168.146.23"
            }
            ]
        },

        "gcware": {
            "nodes": [
            {
                "memberaddr":"192.168.146.22"
            },
            {
                "memberaddr":"192.168.146.23"
            }
            ],
            "client_port":"5919"
        }
}
```

4.2.3.3 GNode Basic Configuration

configuration file

In $GBASE_ BASE/config/gbase_ 8a_ In the gbase.cnf configuration file, you can view and modify the basic configuration of GNode.

In $GBASE_ BASE/config/cluster_ Information such as session timeout is recorded in common.cnf for viewing.

explain

- Unless otherwise specified, all configuration modifications to GNode are made in GBase_ 8a_ In the [gbased] component within gbase.cnf.

Configuration File Format Description

```
[TagName]
Variable_ name = Value
```

Table -46 Parameter Description

Parameter Name	Description
TagName	The component in effect for the variable to be set. ● Client: Client ● Related configurations for Gbased: GNode services

Parameter Name	Description
	● Gbasedump: Related configurations of the gbasedump service ● Gbase: Related configurations of gbase
Variable_ name	Variable name set
Value	Variable value set

Example

GNode configuration file $GBASE_ BASE/config/gbase_ 8a_ The content reference of gbase.cnf is as follows:

```
$ cat $GBASE_ BASE/config/gbase_ 8a_ gbase.cnf
[client]
port=5050
socket = /opt/192.168.146.22/gbase_ 8a_ 5050.sock
#default_ character_ set=gbk



[gbased]
basedir = /opt/192.168.146.22/gnode/server
datadir = /opt/192.168.146.22/gnode/userdata/gbase
skip_ file_ check = 1
bind_ address = 192.168.146.22
socket = /opt/192.168.146.22/gbase_ 8a_ 5050.sock
pid_ file = /opt/192.168.146.22/gnode/log/gbase/gbased.pid

#default_ character_ set=gbk

log_ error
port=5050
#core_ file
gcluster_ node

default_ storage_ engine=express
default_ time_ zone='+8:00'
_ gbase_ query_ path=0
gbase_ parallel_ execution=1

#thread_ cache_ size = 32
```

```
query_ cache_ type = 0
query_ cache_ size = 0M
event_ scheduler=0
skip_ name_ resolve

sql_ mode=PAD_ CHAR_ TO_ FULL_ LENGTH,PIPES_ AS_ CONCAT,ANSI_
QUOTES,IGNORE_ SPACE,NO_ AUTO_ CREATE_ USER,NO_ AUTO_
VALUE_ ON_ ZERO,NO_ ENGINE_ SUBSTITUTION,STRICT_ ALL_
TABLES,NO_ ZERO_ IN_ DATE,NO_ ZERO_ DATE
lower_ case_ table_ names=1

max_ connections = 10000
max_ connect_ errors=1000000
max_ allowed_ packet = 64M
net_ write_ timeout = 1000000
net_ read_ timeout = 1000000
connect_ timeout = 1000000
interactive_ timeout    =1000000
wait_ timeout = 1000000
open_ files_ limit = 65535



#gbase_ memory_ pct_ target=0.8
#gbase_ heap_ data=512M
#gbase_ heap_ temp=256M
#gbase_ heap_ large=256M
#gbase_ buffer_ insert=256M
#gbase_ buffer_ hgrby=10M
#gbase_ buffer_ distgrby=10M
#gbase_ buffer_ hj=10M
#gbase_ buffer_ sj=10M
#gbase_ buffer_ sort=10M
#gbase_ buffer_ rowset=10M
#gbase_ buffer_ result=10M
#gbase_ compression_ sampling=1
gbase_ compression_ str_ method=5
gbase_ compression_ num_ method=5
gbase_ sql_ trace_ level=3
```

```
#gbase_ sql_ trace=1


#enable_ node_ regist = 0


#gbase_ check_ param_ str =



#gbase_ parallel_ max_ thread_ in_ pool
#_ gbase_ parallel_ aggr_ mode=0

thread_ pool_ size = 512
thread_ handling = pool-of-threads

back_ log = 65535
max_ heap_ table_ size = 160000000000
gbase_ express_ log = 1

_ gbase_ optimizer_ in_ subselect = 1

#_ gbase_ enable_ hash_ index_ join = 0
#gbase_ parallel_ auto_ estimate_ optimize = 0
#gbase_ parallel_ auto_ estimate_ number = 16
#gbase_ insertselect_ parallel_ forever = 0

#_ gbase_ enable_ hashtree = 1
#_ gbase_ update_ oneformany    =    1



[gbasedump]
max_ allowed_ packet = 64M



[gbase]
no_ auto_ rehash
```

GNode configuration file $GBASE_ BASE/config/cluster_ The content reference of common.cnf is as follows:

```
{
```

```
    "cluster": {
        "uuid":"a7777256-7bf5-11eb-b80f-000c29b37bff",
        "nodeIPtype":"0",
        "nodes": [
        {
            "memberaddr":"192.168.146.22"
        },
        {
            "memberaddr":"192.168.146.23"
        }
        ]
    },

    "datanode": {
        "localdatanode":"192.168.146.22",
        "sessiontimeout":20,
        "nodes": [
        {
            "memberaddr":"192.168.146.22"
        },
        {
            "memberaddr":"192.168.146.23"
        }
        ]
    },

    "gcware": {
        "nodes": [
        {
            "memberaddr":"192.168.146.22"
        },
        {
            "memberaddr":"192.168.146.23"
        }
        ],
        "client_port":"5919"
    }
}
```

4.2.3.4 GCrecover Basic Configuration

configuration file

On $GCLUSTER_ In the BASE/config/gcmonit.conf configuration file, you can view and modify gc_ Basic configuration of recover.cnf.

Configuration File Format Description

```
$ cat gc_ recover.cnf
[DDL Recovery]
ddl_ recover_ poll_ interval=30
ddl_ recover_ failure_ trick = 1
ddl_ recover_ log_ level=5
ddl_ recover_ gcluster_ connect_ timeout = 100
ddl_ recover_ gcluster_ write_ timeout = 1000000
ddl_ recover_ gcluster_ read_ timeout = 1000000
 [DATA Recovery]
data_ recover_ poll_ interval = 30
data_ recover_ connect_ timeout = 100
data_ recover_ log_ level = 5
data_ recover_ pause=0
data_ recover_ sync_ in_ locking_ stat = 1
recover_ monit_ port = 6268
gcluster_ feventlog_ optimize = 1
```

4.2.3.5 GCmonit Basic Configuration

configuration file

On $GCLUSTER_ The basic configuration of gcmonit can be viewed and modified in the BASE/config/gcmonit.conf configuration file.

Configuration File Format Description

```
[TagName]
fail2ok_ trigger_ cmd=
prog_ name=
ok2fail_ trigger_ cmd=""

[common]
log_ flag=
retry_ times=
interval=
log_ file=""
```

Table -47 Parameter Description

Parameter Name	Description
TagName	The name of the service program to be monitored.
fail2ok_ trigger_ cmd	The command line method that needs to be executed after the monitoring program transitions from stopped to running state. Optional settings, users can set according to their needs. If abnormal settings are found (such as duplicate settings, incorrect settings, etc.), the gcmonit program will report an error and exit.
prog_ name	The specific process name corresponding to the cluster service program has been specified and must be specified in the configuration file. If not specified, the gcmonit program will exit with an error.
ok2fail_ trigger_ cmd	After the monitored program transitions from running to stopped state, or during retry_ The command line method that needs to be executed from stopped to stopped within times. Optional settings, users can set according to their needs. If abnormal settings are found (such as duplicate settings, incorrect settings, etc.), the gcmonit program will report an error and exit.
common	General setting node label, the configuration under this label is for the gcmonit program configuration.
log_ flag	Does the gcmonit program generate a log file during startup. 1 represents generating log information; 0 means no log information is generated. The default value is 1.
retry_ times	The number of consecutive failures of gcmonit to start the monitored program, set to a non-negative integer. The minimum value is 0, representing infinite retries; The maximum value is 64.
interval	The time interval of gcmonit's detection service program, set to a positive integer. The unit is seconds. The minimum value is 1, and the maximum value is 3600.
log_ file	The absolute path of the gcmonit log file. If not specified, the program will report an error and exit.

Example

Configuration file $GCLUSTER for gcmonit_ The content of BASE/config/gcmonit.conf is referenced as follows:

```
$ cat $GCLUSTER_ BASE/config/gcmonit.conf
[gcluster]
fail2ok_ trigger_ cmd=
prog_ name=gclusterd
ok2fail_ trigger_ cmd="/bin/bash
/home/gbase/gcluster/server/bin/gcluster_services gcluster start"

[gcrecover]
fail2ok_ trigger_ cmd=
```

```
prog_ name=gcrecover
ok2fail_ trigger_ cmd="/bin/bash
/home/gbase/gcluster/server/bin/gcluster_services gcrecover start"

[common]
log_ flag=1
retry_ times=10
interval=5
log_ file="/home/gbase/gcluster/log/gcluster/gcmonit.log"

[gcmmonit]
fail2ok_ trigger_ cmd=
prog_ name=gcmmonit
ok2fail_ trigger_ cmd="/home/gbase/gcluster/server/bin/gcmmonit --start"

[gbase]
fail2ok_ trigger_ cmd=
prog_ name=gbased
ok2fail_ trigger_ cmd="/bin/bash
/home/gbase/gcluster/server/bin/gcluster_services gbase start"

[syncserver]
fail2ok_ trigger_ cmd=
prog_ name=gc_ sync_ server
ok2fail_ trigger_ cmd="/bin/bash
/home/gbase/gcluster/server/bin/gcluster_services syncserver start"

[gcware]
fail2ok_ trigger_ cmd=""
ok2fail_ trigger_ cmd="/home/gbase/gcware/sbin/gcware start"
prog_ name=gcware
```

4.2.3.6 GCmmonit Basic Configuration

configuration file

On $GCLUSTER_ The basic configuration of gcmmonit can be viewed and modified in the BASE/config/gcmmonit. conf configuration file.

Configuration File Format Description

```
[common]
log_ file=
log_ flag=
retry_ times=
```

```
interval=

[TagName]
fail2ok_ trigger_ cmd=
ok2fail_ trigger_ cmd=
prog_ name=
```

Table -48 Parameter Description

Parameter Name	Description
common	General setting node label, the configuration under this label is for the gcmonit program configuration.
log_ file	The absolute path of the gcmmonit log file. If not specified, the program will report an error and exit.
interval	The time interval of gcmmonit's detection service program, set to a positive integer. The unit is seconds. The minimum value is 1, and the maximum value is 3600. If it is not specified or exceeds the specified value, the program will exit with an error.
retry_ times	The number of consecutive failures of gcmmonit to start the monitored program, set to a non-negative integer. The minimum value is 0, representing infinite retries; The maximum value is 64. If it is not specified or exceeds the specified value, the program will exit with an error.
log_ flag	Is a log file generated during the startup process of the gcmmonit program. 1 represents generating log information; 0 means no log information is generated. The default value is 1.
TagName	The name of the service program to be monitored.
fail2ok_ trigger_ cmd	The command line method that needs to be executed after the monitoring program transitions from stopped to running state. Optional settings, users can set according to their needs. If abnormal settings are found (such as duplicate settings, incorrect settings, etc.), the gcmonit program will report an error and exit.
ok2fail_ trigger_ cmd	After the monitored program transitions from running to stopped state, or during retry_ The command line method that needs to be executed from stopped to running within times. It is an optional setting item that users can set according to their needs. If abnormal settings are found (such as duplicate settings, incorrect settings, etc.), the gcmonit program will report an error and exit.
prog_ name	The specific process name corresponding to the cluster service program has been specified and must be specified in the configuration file. If not specified, the gcmonit program will exit with an error.

Example

Configuration file $GCLUSTER for gcmmonit_ The content of BASE/config/gcmmonit. conf is referenced as follows:

```
$ cat $GCLUSTER_ BASE/config/gcmmonit.conf
[common]
log_ file="/opt/192.168.146.22/gcluster/log/gcluster/gcmmonit.log"
log_ flag=1
retry_ times=10
interval=5

[gcmonit]
fail2ok_ trigger_ cmd="echo gcmonit started again"
ok2fail_ trigger_ cmd="/opt/192.168.146.22/gcluster/server/bin/gcmonit --start"
prog_ name=gcmonit
```

4.2.3.7 GCware_ Monit Basic Configuration

configuration file

On $GCWARE_ BASE/config/gcware_ In the monit.conf configuration file, you can view and modify gcware_ Basic configuration of monit.

Configuration File Format Description

```
[TagName]
fail2ok_ trigger_ cmd=
prog_ name=
ok2fail_ trigger_ cmd=""

[common]
log_ flag=
retry_ times=
interval=
log_ file=""
```

Table -49 Parameter Description

Parameter Name	Description
TagName	The name of the service program to be monitored.
fail2ok_ trigger_ cmd	The command line method that needs to be executed after the monitoring program transitions from stopped to running state. Optional settings, users can set according to their needs. If abnormal settings are found (such as duplicate settings, incorrect settings, etc.), gcware_ The monit program will exit with an error.
prog_ name	The specific process name corresponding to the service program has been specified and must be specified in the configuration file.

Parameter Name	Description
	If not specified, gcware_ The monit program will exit with an error.
ok2fail_ trigger_ cmd	After the monitored program transitions from running to stopped state, or during retry_ The command line method that needs to be executed from stopped to stopped within times. Optional settings, users can set according to their needs. If abnormal settings are found (such as duplicate settings, incorrect settings, etc.), gcware_ The monit program will exit with an error.
common	General setting node label, the configuration under this label is gcware_ Monit program configuration.
log_ flag	gcware_ Whether to generate a log file during the startup process of the monit program. 1 represents generating log information; 0 means no log information is generated. The default value is 1.
retry_ times	gcware_ The number of consecutive failures of monit to start the monitored program, set to a non-negative integer. The minimum value is 0, representing infinite retries; The maximum value is 64.
interval	gcware_ The time interval of the monit detection service program, set to a positive integer. The unit is seconds. The minimum value is 1, and the maximum value is 3600.
log_ file	gcware_ The absolute path of the monit log file. If not specified, the program will exit with an error.

Example

gcware_ Monit's configuration file $GCWARE_ BASE/config/gcware_ The content reference of monit.conf is as follows:

```
$ cat $GCWARE_ BASE/config/gcware_ monit.conf
[gcware]
fail2ok_ trigger_ cmd=""
prog_ name=gcware
ok2fail_ trigger_ cmd="/opt/192.168.146.22/gcware/sbin/gcware start"

[common]
log_ flag=1
retry_ times=10
interval=5
log_ file="/opt/192.168.146.22/gcware/log/gcware_monit.log"

[gcware_mmonit]
fail2ok_ trigger_ cmd=""
prog_ name=gcware_ mmonit
ok2fail_ trigger_ cmd="/opt/192.168.146.22/gcware/sbin/gcware_mmonit --start"
```

4.2.3.8 GCware_ Basic configuration of mmonit

configuration file

On $GCWARE_ BASE/config/gcware_ In the mmonit.cnf configuration file, you can view and modify gcware_ The basic configuration of mmonit.

Configuration File Format Description

```
[common]
log_ file=
log_ flag=
retry_ times=
interval=

[TagName]
fail2ok_ trigger_ cmd=
ok2fail_ trigger_ cmd=
prog_ name=
```

Table -410 Parameter Description

Parameter Name	Description
common	General setting node label, the configuration under this label is gcware_ MMONIT program configuration.
log_ file	gcware_ The absolute path of the mmonit log file. If not specified, the program will report an error and exit.
interval	gcware_ The time interval for the detection service program of mmonit, set to a positive integer. The unit is seconds. The minimum value is 1, and the maximum value is 3600. If it is not specified or exceeds the specified value, the program will exit with an error.
retry_ times	gcware_ The number of consecutive failures of mmonit to start the monitored program, set to a non-negative integer. The minimum value is 0, representing infinite retries; The maximum value is 64. If it is not specified or exceeds the specified value, the program will exit with an error.
log_ flag	gcware_ Is a log file generated during the startup process of the mmonit program. 1 represents generating log information; 0 means no log information is generated. The default value is 1.
TagName	The name of the service program to be monitored.
fail2ok_ trigger_ cmd	The command line method that needs to be executed after the monitoring program transitions from stopped to running state. Optional settings, users can set according to their needs. If abnormal settings are found (such as duplicate settings, incorrect

Parameter Name	Description
	settings, etc.), gcware_ The mmonit program will exit with an error.
ok2fail_ trigger_ cmd	After the monitored program transitions from running to stopped state, or during retry_ The command line method that needs to be executed from stopped to running within times. It is an optional setting item that users can set according to their needs. If abnormal settings are found (such as duplicate settings, incorrect settings, etc.), gcware_ The mmonit program will exit with an error.
prog_ name	The specific process name corresponding to the cluster service program has been specified and must be specified in the configuration file. If not specified, gcware_ The mmonit program will exit with an error.

Example

gcware_ Configuration file $GCWARE for mmonit_ BASE/config/gcware_ The reference for the content of mmonit. conf is as follows:

```
$ cat $GCWARE_ BASE/config/gcware_ mmonit.conf
[gcware_monit]
fail2ok_ trigger_ cmd="echo gcware monit started again"
prog_ name=gcware_ monit
ok2fail_ trigger_ cmd="/opt/192.168.146.22/gcware/sbin/gcware_monit --start"

[common]
log_ flag=1
retry_ times=10
interval=5
log_ file="/opt/192.168.146.22/gcware/log/gcware_mmonit.log"
```

4.2.3.9 Gcware monitors gnode status related configurations through registration

Gnode registration mechanism

After gcware separation, it is possible to enable gnode registration with gcware through parameter control, allowing gcware to monitor the status of gnode through registration. Registration information includes:

- The vcid to which gnode belongs
- Global consistency parameters are required for node registration in gnode, and currently only the following three parameters are involved:

gbase_ segment_ size

gbase_ compression_ str_ method、

gbase_ compression_ num_ method

- Correspondence between the session ID and nodeid of the connection between the gnode node and gcware

The above registration information is used for:

1. Check whether the global consistency parameters of gnode are consistent within the same VC, and this function is controlled by the parameters.

Gcware unconditionally retained the global consistency parameters of the first registered gnode. All the global consistency parameters provided during subsequent gnode registration were compared with the global consistency parameters retained by gcware, and the comparison was consistent. The gnode registration was successful, but the comparison was not consistent. gcware directly exited, believing that the gnode did not exist. The parameter detection status can be saved in the system. log of gnode and the gcware. log of gcware.

2. Track the status of gnode nodes through the heartbeat keeping mechanism corresponding to the registered session ID, which is controlled by parameters. If the heartbeat is interrupted, gcware detects a session timeout and sets the ignore status to be abnormal.

3. The registration information will be recorded in the log.

The system log of gnode will record the global consistency parameter information returned by gcware; If the global consistency parameter detection of gcware is inconsistent, the parameter information will be recorded in the gcware. log of the gcware node

Control parameters

- Gnode configuration file gbase_ 8a_ gbase.cnf
  - enable_ node_ regist

Whether to enable the registration mechanism. The default value is 0, which means it is not enabled, and 1, which means it is enabled.

  - gbase_ check_ param_ str

The parameters that need to be detected for parameter consistency detection, with a default value of:

gbase_ segment_ size%gbse_ compression_ str_ method%gbase_ compression_ num_ method


explain

enable_ node_ When register is 1, gnode will only perform consistency check and read gbase when it is started_ check_ param_ The content in str. For situations where older versions are upgraded, the registration mechanism is not enabled by default. To enable it, you need to manually fill in these two parameters.

- Gcware configuration file gcware. conf
  - enable_ node_ regist

Whether to enable the registration mechanism. The default value is 0, not enabled, and enabled when set to 1. After enabling, gnode needs to be registered with gcware in order for gcware to consider it online, otherwise gcware considers it offline. After gcware enables this parameter, all gnodes under VC need to be registered with gcware.

  - enable_ check_ param

Whether to enable the parameter consistency detection mechanism. The default value is 0 and not enabled. When set to 1, it means enabled.

- Gcluster configuration file gbase_ 8a_ gcluster.cnf
  - enable_ node_ regist

When gcluster fails to execute SQL and returns a loss connection, does gcluster set the gnode node service status to offline. The default value is 0, which sets the service status of the gnode node to offline. When set to 1, it does not set the service status of the gnode node to offline.

be careful

- By default, gcware monitors the status of gnode through the registration mechanism, but gnode status detection still uses the original mechanism:

  Detect gnode status through timed detection, execution of gcadmin, and successful execution of SQL

- If you need to enable the gnode registration mechanism, please note in parameter settings that gnode, gcware, and gccluster all have their own parameters for enabling the registration mechanism, which have their own meanings and need to be correctly combined to make sense.

- Attention should be paid to the parameter matching when the registration mechanism is enabled:
  1) It can enable ignore, gcluster, and gcware_ node_ Register can be enabled simultaneously.
  2) You can choose to enable both gnode and gcware_ node_ Register is enabled simultaneously, and after enabling it, you can choose whether to enable gcware or not_ check_ param;
  3) You can choose to enable gcluster and gcware_ node_ Register and enable a single service
  separately_ node_ Registering is meaningless. Only when the gnode registration mechanism parameters are turned on can registration information exist, and only when the gcware registration mechanism parameters are turned on and consistency checks are meaningful. If the gnode registration mechanism is turned off, opening the gcware registration mechanism and consistency checks is meaningless. vice versa.

- If the consistency detection mechanism is enabled and global consistency parameters are modified, all gnodes in VC need to stop the process modification.

- The gcadmin createVC and gcadmin distribution commands also perform global consistency parameter detection on all gnodes within the VC.

- The default timeout for GCware's session heartbeat detection on gnode is election_ Timeout * 20, in seconds, election_ The timeout is configured in gcware. conf. The heartbeat timeout can also be found in the gnode configuration file cluster_ The session timeout setting in the datanode module of commonconf cannot be less than 6 seconds.

4.3 Cluster management

4.3.1 gcadmin

4.3.1.1 Distribution Management Command

4.3.1.1.1 Distribution command

function

After installing the cluster and generating the distribution, you need to use this command to formulate the distribution strategy for node sharding.

be careful

- This command needs to be switched to dbaUser for proper execution. If using another user to execute the generate distribution command, the user will be prompted to switch to dbaUser to execute the command, and an error will be reported to exit;
- If you use single VC mode (compatibility mode) to install a cluster, after the installation is completed, a distribution will be generated directly, and all free nodes will be added to the default VC;
- If a new gnode node is expanded without generating a distribution, the operations of creating database users and modifying database passwords performed before the expansion will not take effect on the newly expanded gnode node. Instead, the database users and modifying passwords must be recreated after the expansion.

There are three distribution configurations: load balancing mode, high availability mode, and custom sharding distribution mode (custom distribution mode). If not set, it defaults to load balancing mode.

Load balancing mode

Configuration pattern 1 indicates the use of load balancing mode. In this mode, the nodes in each rack in gcChangeInfo.xml are grouped. The backup shard 1 of the main shard on each rack node is stored on the node in the next rack in gcChangeInfo.xml, and the

backup shard 2 is stored on the node in the previous rack in gcChangeInfo.xml. The previous track of the first track in gcChangeInfo.xml is the last track, and the next track of the last track is the first track.

 **explain**

Generate distribution using load balancing mode

- The number of main shards per node (i.e. parameter p) must be less than the number of nodes per rack to ensure uniform distribution of backup shards.
- Each rack should contain the same number of nodes as much as possible. If there is more than one rack in the gcChangeInfo.xml file with a different number of nodes compared to other racks, gcadmin will prompt the user that system performance may decrease and require confirmation from the user before generating a distribution.

High availability mode

The pattern 2 mode is a high availability mode, and the distribution generated in this mode stores the backup shard 1 of each data node on the next data node, while the backup shard 2 is stored on the previous data node. When using high availability mode, only one rack is required in the configuration file gcChangeInfo.xml, and even if there are multiple racks, they will be processed as one rack.

Custom sharding distribution mode

To customize the shard distribution mode, you need to manually write an XML file to configure the distribution shard distribution information, that is, specify the corresponding nodes of each shard's primary and backup shards in the file. Using this method to configure the distribution method does not require the input parameters p, d, and pattern.

grammar

```
gcadmin distribution <gcChangeInfo.xml> <p num> [d num] [extension] [pattern 1|2][db_user user_name] [db_pwd password] [vc vc_name]
```

Table -411 Parameter Description

Parameter Name	explain
gcChangeInfo.xml	Generate a gnode node information file for distribution. After the cluster installation is successful, a sample file named gcChangeInfo.xml will be generated in the directory where the installation package is located. This file is in XML format, with a

Parameter Name	explain
	root tag of<servers>, which describes the data node information for generating distribution; The sub tag is<rack>, which describes the corresponding relationship between the rack and the gnode node. After installation, there is only one<rack>in the generated gcChangeInfo.xml, which contains information about all data nodes in the cluster. When using the pattern 1 mode to generate a distribution, multiple<rack>tags can be inserted according to the machine deployment situation, and the data node information can be inserted under the corresponding<rack>tags.
p num	The minimum number of shards stored in each data node is 1. The p-value multiplied by the number of data nodes should not exceed 65535, which means the total number of shards in the cluster should not exceed 65535. Otherwise, gcadmin will report an error and exit.
d num	The number of backups per shard, with values of 0, 1, or 2. When the value is 0, user confirmation is required; If the parameter d is not entered, the default value is 1. When the number of nodes cannot meet the backup sharding placement point, an error will be reported.
pattern 1|2	The mode used to generate the distribution, with a number value of 1 or 2, pattern 1 being the load balancing mode, and pattern 2 being the high availability mode. If the parameter pattern is not entered, the distribution will be generated using pattern 1 by default.
extension	The generated new distribution will distribute the original shards as much as possible on the original nodes.
db_ user user_ name	When expanding to generate a new distribution, the database username needs to be passed in. Not specified, the default database user is root. After expansion, create a new virtual cluster, and pass in this parameter when generating the distribution of the virtual cluster in the newly generated virtual cluster.
db_ pwd password	If db_ user user_ The name specifies that the user's password is not empty, and when generating a new distribution, the user's password needs to be passed in during command execution. Special characters in passwords need to be added with escape character. When the

Parameter Name	explain
	password is empty, this parameter does not need to be passed in, only the database username needs to be passed in.
vc vc_ name	Specify the name of the virtual cluster that generates the distribution. If the cluster has only one VC, generating distribution does not require specifying vcname

Example

Example 1: Load balancing mode setting.

After installing the cluster, a gcChangeInfo.xml file containing all data node information will be generated in the installation package directory on the node performing the installation operation, as shown below:

```
$ cat gcChangeInfo.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
 <rack>
   <node ip="192.168.153.128"/>
   <node ip="192.168.153.129"/>
   <node ip="192.168.153.133"/>
   <node ip="192.168.153.134"/>
   <node ip="192.168.153.130"/>
   <node ip="192.168.153.137"/>
 </rack>
</servers>
```

Based on the actual rack and machine deployment situation, insert the<rack>tag in the file as follows:

```
<? xml version="1.0" encoding="utf-8"?>
<servers>
      <rack>
            <node ip="192.168.153.128"/>
            <node ip="192.168.153.129"/>
      </rack>
      <rack>
            <node ip="192.168.153.133"/>
            <node ip="192.168.153.134"/>
       </rack>
      <rack>
            <node ip="192.168.153.130"/>
            <node ip="192.168.153.137"/>
       </rack>
</servers>
```

The command reference for generating distribution using load balancing mode and modified gcChangeInfo.xml file is as follows:

```
$ gcadmin distribution gcChangeInfo.xml p 1 d 2 pattern 1

gcadmin generate distribution ...
```

```
NOTE: node [192.168.153.129] is coordinator node, it shall be data node too

NOTE: node [192.168.153.130] is coordinator node, it shall be data node too

gcadmin generate distribution successful
```

After generating a distribution, you can use gcadmin showdistribution to view the distribution information of the primary and secondary nodes of the data, as shown below:

```
$ gcadmin showdistribution


                Distribution ID: 1 | State: new | Total segment num: 6


     Primary Segment Node IP          Segment ID          Duplicate
Segment node IP
=============================================================
==========================
|        192.168.153.128        |        1          |
192.168.153.134                                      |
192.168.153.137
-------------------------------------------------------------------------------
|        192.168.153.129        |        2          |
192.168.153.133                                      |
192.168.153.130
-------------------------------------------------------------------------------
|        192.168.153.133        |        3          |
192.168.153.137                                      |
192.168.153.129
-------------------------------------------------------------------------------
|        192.168.153.134        |        4          |
192.168.153.130                                      |
192.168.153.128
-------------------------------------------------------------------------------
```

```
|                192.168.153.130                |                5                |
192.168.153.129                                                                 |
192.168.153.134

--------------------------------------------------------------------------------

|                192.168.153.137                |                6                |
192.168.153.128                                                                 |
192.168.153.133

================================================================
========================
```

Use the gcadmin showdistribution node command to view the generated distribution information, as shown below:

```
$ gcadmin showdistribution node


          Distribution ID: 1 | State: new | Total segment num: 5


============================================================
==========================================

|nodes              |  192.168.153.128  |  192.168.153.129  |  192.168.153.133  |
192.168.153.134 | 192.168.153.130 | 192.168.153.137 |

------------------------------------------------------------------------------------------
--

|primary    |          1          |          2          |          3          |
4          |          5          |          6          |

|segments   |                     |                     |                     |
|                     |                     |

------------------------------------------------------------------------------------------
--

|duplicate |          6          |          5          |          2          |
1          |          4          |          3          |

|segments 1|                     |                     |                     |
|                     |                     |
```

```
-------------------------------------------------------------------------------------------------
--
|duplicate |            4            |            3            |            6            |
5            |            2            |            1            |
|segments 2|                         |                         |                         |
|                         |                         |
=============================================================
=========================================
```

Example 2: High availability mode configuration.

Generate a configuration file for distribution, as shown below:

```
<? xml version="1.0" encoding="utf-8"?>
<servers>

    <rack>
        <node ip="192.168.153.128"/>
        <node ip="192.168.153.129"/>
        <node ip="192.168.153.133"/>
    </rack>
    <rack>
        <node ip="192.168.153.134"/>
        <node ip="192.168.153.130"/>
        <node ip="192.168.153.137"/>
      </rack>

</servers>
```

The command reference for generating distribution using high availability mode and modified configuration files is as follows:

```
$ gcadmin distribution gcChangeInfo.xml p 2 d 2 pattern 2
gcadmin generate distribution ...
```

```
gcadmin generate distribution successful
```

After generating a distribution, you can use gcadmin showdistribution to view the partition distribution information of the primary and backup data, as shown below:

```
$ gcadmin showdistribution

                Distribution ID: 21 | State: new | Total segment num: 12

     Primary Segment Node IP          Segment ID          Duplicate Segment node IP
==========================================================================
===============
|       192.168.153.128       |       1       |       192.168.153.129       |
192.168.153.137
--------------------------------------------------------------------------------
|       192.168.153.129       |       2       |       192.168.153.133       |
192.168.153.128
--------------------------------------------------------------------------------
|       192.168.153.133       |       3       |       192.168.153.134       |
192.168.153.129
--------------------------------------------------------------------------------
|       192.168.153.134       |       4       |       192.168.153.130       |
192.168.153.133
--------------------------------------------------------------------------------
|       192.168.153.130       |       5       |       192.168.153.137       |
192.168.153.134
--------------------------------------------------------------------------------
|       192.168.153.137       |       6       |       192.168.153.128       |
192.168.153.130
--------------------------------------------------------------------------------
```

```
| 192.168.153.128 | 7 | 192.168.153.129 |
192.168.153.137
----------
| 192.168.153.129 | 8 | 192.168.153.133 |
192.168.153.128
----------
| 192.168.153.133 | 9 | 192.168.153.134 |
192.168.153.129
----------
| 192.168.153.134 | 10 | 192.168.153.130 |
192.168.153.133
----------
| 192.168.153.130 | 11 | 192.168.153.137 |
192.168.153.134
----------
| 192.168.153.137 | 12 | 192.168.153.128 |
192.168.153.130
================================================================
==============
```

The gcadmin showdistribution node command displays the generated distribution information as follows:

```
$ gcadmin showdistribution node

                    Distribution ID: 21 | State: new | Total segment num: 12

================================================================
================================
|nodes | 192.168.153.128 | 192.168.153.129 | 192.168.153.133 | 192.168.153.134 |
192.168.153.130 | 192.168.153.137 |
----------
|primary | 1 | 2 | 3 | 4
```

```
|          5          |          6          |
|segments   |          7          |          8          |          9          |          10
|          11          |          12          |
-----------------------------------------------------------------------------------------------
|duplicate |          6          |          1          |          2          |          3          |
4          |          5          |
|segments 1|          12          |          7          |          8          |          9
|          10          |          11          |
-----------------------------------------------------------------------------------------------
|duplicate |          2          |          3          |          4          |          5          |
6          |          1          |
|segments 2|          8          |          9          |          10          |          11
|          12          |          7          |
================================================================================
================================
```

Example 3: Custom sharding distribution mode configuration.

Generate the gcChangeInfo.xml file for distribution, as shown below:

```
<? xml version="1.0" encoding="utf-8"?>
<servers>

    <cfgFile file="distribution.xml"/>

</servers>
```

The distribution sharding configuration information file, distribution.xml, is as follows:

```
<? xml version='1.0' encoding="utf-8"?>
<distributions>
  <distribution>
      <segments>
```

```
            <segment>
                <primarynode ip="192.168.153.125"/>

                <duplicatenodes>
                    <duplicatenode ip="192.168.153.126"/>
                    <duplicatenode ip="192.168.153.137"/>
                </duplicatenodes>
            </segment>

            <segment>
                <primarynode ip="192.168.153.128"/>

                <duplicatenodes>
                    <duplicatenode ip="192.168.153.129"/>
                </duplicatenodes>
            </segment>

        </segments>
    </distribution>
</distributions>
```

The custom partition distribution mode generates a distribution as follows:

```
$ gcadmin distribution gcChangeInfo.xml
gcadmin generate distribution ...

gcadmin generate distribution successful
```

After generating a distribution, you can use the gcadmin showdistribution command to view the generated distribution information, as shown below:

```
$ gcadmin showdistribution

                Distribution ID: 3 | State: new | Total segment num: 2

     Primary Segment Node IP          Segment ID          Duplicate Segment node IP
=============================================================================
===============
|       192.168.153.125       |       1       |       192.168.153.126       |
|                             |               |       192.168.153.137       |
|
-----------------------------------------------------------------------------
|       192.168.153.128       |       2       |       192.168.153.129       |
=============================================================================
===============
```

4.3.1.1.2 **Rmdistribution command**

function

Remove the distribution with the specified id from the cluster. If the distribution ID is not entered, the distribution with an earlier creation time will be deleted by default. If there is only one distribution in the cluster, the distribution will be deleted by default.

be careful

- When deleting a distribution, it is necessary to first confirm that all GCluster nodes are functioning properly. If there is a fevent log in the distribution, it must be cleared before deleting the distribution. If the GCluster node service is abnormal, it will result in unrecoverable fevent logs after deleting the distribution;
- If the distribution to be deleted has ddl event, dml event, or dmlstorage event, the fevent log must be cleared before deleting the distribution. Otherwise, gcadmin will report an error and exit;
- This command needs to be switched to DBA User for execution, otherwise gcadmin will prompt the switching user to execute the command and report an error to exit;
- If GC_ stats_ Table and GC_ stats_ The column table uses the distribution ID that will be deleted, so the user first needs to delete the GC_ stats_ Table and GC_ stats_ Rebalance the two tables in column to another distribution ID, and then perform refreshnodatamap and delete distribution operations.

grammar

```
gcadmin rmdistribution [ID] [vc vc_name]
```

Table -412 Parameter Description

Parameter Name	explain
ID	distribution id。
vc vc_ name	Specify the name of the VC for the deleted distribution. If the cluster has only one VC, there is no need to specify a VCname.

Example

```
$ gcadmin rmdistribution 1 vc vc1
distribution: id [1] is current distribution
it will be removed now
please ensure this is ok, input y or n: y
gcadmin remove distribution [1] success
```

4.3.1.1.3 Showdistribution command

function

Display distribution information.

 **explain**

- The IP order displayed by executing the gcadmin showdistribution command is independent of the IP order of the configuration file that generated the distribution.

grammar

```
gcadmin showdistribution [node | f] [vc vc_name]
```

Table -413 Parameter Description

Parameter Name	explain
node	Display the primary/secondary partitions contained by each node in the distribution, with optional parameters.
f	Display information in XML format
vc vc_ name	The name of the VC to display distribution information.

Example

Generate using the gcadmin distribution gcChangeInfo.xml p 2 d 2 pattern 1 command

The distribution, gcChangeInfo.xml, is as follows:

```
<? xml version="1.0" encoding="utf-8"?>
<servers>
    <rack>
        <node ip="192.168.153.128"/>
        <node ip="192.168.153.129"/>
        <node ip="192.168.153.133"/>
    </rack>
    <rack>
        <node ip="192.168.153.134"/>
        <node ip="192.168.153.130"/>
          <node ip="192.168.153.137"/>
      </rack>
</servers>
```

After generating distribution, use the command gcadmin showdistribution without entering the parameter node to display distribution information in sharded order, as shown below:

```
$ gcadmin showdistribution
```

```
Distribution ID: 24 | State: new | Total segment num: 12

   Primary Segment Node IP          Segment ID          Duplicate
Segment node IP
============================================================
==========================
|          192.168.153.128          |          1          |
192.168.153.129                                          |
192.168.153.137
----------------------------------------------------------------------------
|          192.168.153.129          |          2          |
192.168.153.133                                          |
192.168.153.128
----------------------------------------------------------------------------
|          192.168.153.133          |          3          |
192.168.153.134                                          |
192.168.153.129
----------------------------------------------------------------------------
|          192.168.153.134          |          4          |
192.168.153.130                                          |
192.168.153.133
----------------------------------------------------------------------------
|          192.168.153.130          |          5          |
192.168.153.137                                          |
192.168.153.134
----------------------------------------------------------------------------
|          192.168.153.137          |          6          |
192.168.153.128                                          |
192.168.153.130
```

```
--------------------------------------------------------------------------------

|              192.168.153.128              |              7               |
192.168.153.129                                                              |
192.168.153.137

--------------------------------------------------------------------------------

|              192.168.153.129              |              8               |
192.168.153.133                                                              |
192.168.153.128

--------------------------------------------------------------------------------

|              192.168.153.133              |              9               |
192.168.153.134                                                              |
192.168.153.129

--------------------------------------------------------------------------------

|              192.168.153.134              |              10              |
192.168.153.130                                                              |
192.168.153.133

--------------------------------------------------------------------------------

|              192.168.153.130              |              11              |
192.168.153.137                                                              |
192.168.153.134

--------------------------------------------------------------------------------

|              192.168.153.137              |              12              |
192.168.153.128                                                              |
192.168.153.130
================================================================================
=========================
```

Use the command gcadmin showdistribution node to display distribution shards by node

The information is as follows:

```
$ gcadmin showdistribution node



                                  Distribution ID: 24 | State: new | Total segment num:
12
```

```
============================================================
==========================================
|nodes              | 192.168.153.128 | 192.168.153.129 | 192.168.153.133 |
192.168.153.134 | 192.168.153.130 | 192.168.153.137 |
------------------------------------------------------------------------------------------
--
|primary    |          1          |          2          |          3          |
4           |          5          |          6          |
|segments   |          7          |          8          |          9          |
10          |          11         |          12         |
------------------------------------------------------------------------------------------
--
|duplicate |          6          |          4          |          5          |
3           |          1          |          2          |
|segments 1|          11         |          12         |          10         |
8           |          9          |          7          |
------------------------------------------------------------------------------------------
--
|duplicate |          2          |          3          |          1          |
5           |          6          |          4          |
|segments 2|          9          |          7          |          8          |
12          |          10         |          11         |
============================================================
==========================================
```

4.3.1.1.4 **Getdistribution command**

function

Save the distribution information of the specified ID to the specified file, and the generated file is an XML file. Users can modify the sharding information in this file, and then use this file to regenerate the distribution.

be careful

The specified distribution must be an existing distribution. If the file specified for storing distribution information already exists, its content will be cleared and new distribution sharding information will be written.

grammar

```
gcadmin getdistribution <ID> <file_ name.xml> [vc vc_name]
```

Table -414 Parameter Description

Parameter Name	explain
ID	The distribution id to obtain.
file_ name.xml	The file name for saving distribution information.
vc vcname	Specify the name of the VC to obtain distribution information.

Example

Execute command:

```
$ gcadmin getdistribution 6 dstb_ info

gcadmin getdistribution 6 dstb_ info ...



get segments information

write segments information to file [dstb_info]



gcadmin getdistribution information successful
```

Command execution successful, generated dstb_ The content of the info file is as follows:

```
<? xml version='1.0' encoding="utf-8"?>

<distributions>

    <distribution>

    <segments>

        <segments>
```

```
                <primarynode ip=”192.168.153.129”/>


                    <duplicatenodes>
                        <duplicatenode ip=”192.168.153.125”/>
                        <duplicatenode ip=”192.168.153.126”/>
                    </duplicatenodes>
            </segments>

        <segments>
                <primarynode ip=”192.168.153.125”/>


                    <duplicatenodes>
                        <duplicatenode ip=”192.168.153.129”/>
                    </duplicatenodes>
            </segments>
</segments>
        </distribution>
</distributions>
```

4.3.1.2 Node management commands

4.3.1.2.1 **Addnodes command**

function

Add the specified data node in gcChangeInfo.xml to VC or RC. After the cluster installation is successful, this command will be automatically called to add the successfully installed data nodes to the cluster without the need for the user to manually execute the addnodes command. After successful execution, calling this command again will result in an error and exit, indicating that the user node has been added to the cluster.

be careful

This command is an internal system command that will be automatically called by the system after installing the cluster. It is not recommended for users to use.

grammar

```
gcadmin addnodes gcChangeInfo.xml [vc_name | single_vc_add_to_rc]
```

Table -415 Parameter Description

Parameter Name	explain
gcChangeInfo.xml	GcChangeInfo.xml is the data node information to be added, which only needs to contain one<rack>. Using multiple racks to specify multiple node information has the same effect as specifying all node information with one rack. After the cluster installation is successful, this file will be generated and used to automatically call the addnodes command.
vc_ name	Specify the VC name to add the node to the VC.
single_ vc_ add_ to_ rc	In compatibility mode, add the new node to RootCluster as a freenode.

Example

Step 1: Modify the gcChangeInfo.xml file:

```
$ cat gcChangeInfo.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
  <rack>
    <node ip="172.168.83.15"/>
  </rack>
</servers>
```

Step 2: Add freenode to vc1:

```
$ gcadmin addnodes gcChangeInfo.xml vc1
gcadmin add nodes ...

flush statemachine success

gcadmin addnodes to vc [vc1] success
After adding, the cluster status information is as follows:
$ gcadmin showcluster vc vc1
CLUSTER STATE:              ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL
```

```
=============================================
|          GBASE VIRTUAL CLUSTER INFORMATION          |
=============================================
|       VcName       | DistributionId |     comment     |
---------------------------------------------
|        vc1         |            1            | vc1comments |
---------------------------------------------
=============================================================================
======
|                                        VIRTUAL CLUSTER DATA NODE INFORMATION
|
=============================================================================
======
| NodeName |    IpAddress          | DistributionId | gnode | syncserver | DataState |
-----------------------------------------------------------------------------
|   node1    |172.168.83.11      |           1           | OPEN  |     OPEN      |       0       |
-----------------------------------------------------------------------------
|   node2    |172.168.83.12      |           1           | OPEN  |     OPEN      |       0       |
-----------------------------------------------------------------------------
|   node3    |172.168.83.15      |                       | OPEN  |     OPEN      |       0       |
3 data node
```

4.3.1.2.2 **Rmnodes command**

function

Remove the data node specified in gcChangeInfo.xml from the cluster.

grammar

```
gcadmin rmnodes gcChangeInfo.xml [vc_name | single_vc_rm_to_rc]
Freenode can only be removed without optional parameters
gcadmin rmnodes gcChangeInfo.xml
```

Table -416 Parameter Description

Parameter Name	explain
gcChangeInfo.xml	GcChangeInfo.xml is the data node information to be deleted.
vc_ name	Specify VC_ Name will remove the node corresponding to gcChangeInfo.xml in the VC and change it to a free node in the root cluster.

Parameter Name	explain
single_ vc_ rm_ to_ rc	Remove nodes from the cluster in compatibility mode, that is, remove nodes from the default VC to the root cluster and become free nodes.

 **be careful**

1. The data node to be deleted must be a data node that has been added to the current cluster and has not been used by any distribution.

2. Removing empty nodes (nodes that have been cleared of data after redistribution) from the specified VC from the entire cluster requires two steps:

Remove the nodes specified in gcChangeInfo.xml from VC and change them to freenodegsadmin
rmnodes of the cluster gcChangeInfo.xml VC_ name;

Completely remove
gcadmin rmnodes gcChangeInfo.xml from the cluster

Example

Step 1: Modify gcRm_ VC1. XML, where the corresponding nodeIP content is the node IP to be deleted:

```
$ cp gcChangeInfo.xml gcRm_ vc1.xml
$ vi gcRm_ vc1.xml
$ cat gcRm_ vc1.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
  <rack>
    <node ip="192.168.146.40"/>
  </rack>
</servers>
```

Step 2: Delete the node from the corresponding VC

```
$ gcadmin rmnodes gcRm_ vc1.xml vc1
gcadmin remove nodes ...

flush statemachine success

gcadmin rmnodes from vc [vc1] success

$ gcadmin
CLUSTER STATE:                ACTIVE


=======================================
|   GBASE GCWARE CLUSTER INFORMATION   |
=======================================
| NodeName |      IpAddress       | gcware |
---------------------------------------
| gcware1   | 192.168.146.20 |   OPEN   |
---------------------------------------
| gcware2   | 192.168.146.21 |   OPEN   |
---------------------------------------
| gcware3   | 192.168.146.22 |   OPEN   |
---------------------------------------
=============================================================
|              GBASE COORDINATOR CLUSTER INFORMATION               |
=============================================================
|     NodeName     |      IpAddress       | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 192.168.146.20 |     OPEN     |         0         |
-------------------------------------------------------------
| coordinator2 | 192.168.146.22 |     OPEN     |         0         |
-------------------------------------------------------------
=============================================
|        GBASE VIRTUAL CLUSTER INFORMATION         |
=============================================
|       VcName       | DistributionId |   comment   |
---------------------------------------------
|         vc1          |            3             | vc1            |
---------------------------------------------
=================================================================
|                GBASE CLUSTER FREE DATA NODE INFORMATION                  |
=================================================================
| NodeName    |      IpAddress       | gnode | syncserver | DataState |
```

4.3.1.2.3 Switchmode command

function

Switch the cluster state to normal, readonly, or recovery.

explain

- When performing cluster level backups, it is necessary to set the status of all VCs in the cluster to readonly;
- When performing cluster level recovery, it is necessary to set all VC states in the cluster to the recovery state.

grammar

```
gcadmin switchmode <mode> <vc vc_ name | coordinator>
```

Table 417 Parameter Description

Parameter Name	explain
mode	Mode contains three states: normal, readonly, and recovery. 1、Normal indicates that the cluster is in a usable and normal state. 2、Readonly indicates that the cluster is in a read-only state, and only query related operations can be performed. During node replacement and gcrcman backup, the cluster is changed to a read-only state. 3、Recovery indicates that the cluster is in a recovery state and needs to be changed to a recovery state before gcrcman can be restored.
vc vc_ name	Set the virtual cluster name for the cluster state
coordinator	Setting the state of the management cluster is currently only used for node replacement internal calls, and this parameter is invalid in other scenarios. It is not recommended for users to manually execute this parameter.

Example

View cluster status:

```
$ gcadmin showcluster vc vc1
CLUSTER STATE:              ACTIVE
VIRTUAL CLUSTER MODE:   RECOVERY
```

```
============================================================
|                  GBASE VIRTUAL CLUSTER INFORMATION                  |
============================================================
|     VcName       | DistributionId |                comment                |
------------------------------------------------------------
|       vc1        |              1              | comment message for vc1 |
------------------------------------------------------------
============================================================
=======
|                    VIRTUAL CLUSTER DATA NODE INFORMATION
|
============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
---------------------------------------------------------------------
| node1    |172.168.83.11|           1             |OPEN |     OPEN     |       0       |
---------------------------------------------------------------------
| node2    |172.168.83.12|           1             |OPEN |     OPEN     |       0       |
---------------------------------------------------------------------

2 data node
After the backup is completed, set the cluster state to normal:
$ gcadmin switchmode normal vc vc1

========== switch cluster mode...
   switch pre mode:                        [READONLY]
   switch mode to                          [NORMAL]
   switch after mode:                      [NORMAL]
```

4.3.1.2.4 Setnodestate command

function

Set the state of a node.

grammar

```
gcadmin setnodestate ip <state>
```

Table -418 Parameter Description

Parameter Name	explain
ip	Node IP to set status.

Parameter Name	explain
state	Used to specify the node state to be set, there are three types of node states. ● Unavailable identifies a node as unavailable and does not record dml or ddl operations on that node. After setting this state, node replacement must be performed. After the replacement is completed, the state can be restored to normal; ● Failure: Identifies a cluster failure, which is equivalent to offline. At this time, dml and ddl will not be distributed to the node, but will be directly recorded as a fault log; ● Normal: After the node fault is resolved, the node can be directly set to normal, which is equivalent to the node being reconnected. At this time, gcrecover will restore the previously recorded feventlog, and newly initiated ddl and dml will be redistributed to the node.

 **warning**

- Once the node status is set to unavailable, it cannot be manually restored and can only be replaced for availability;
- If setting a node to unavailable status will result in the primary and secondary partitions of a partition being unavailable in any distribution, then the setting fails.

Example

```
$ gcadmin setnodestate 172.168.83.13 failure
set node [172.168.83.13] state to failure
set node [172.168.83.13] state to failure successful

$ gcadmin showcluster vc vc2
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

============================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                    |
============================================================
|     VcName       | DistributionId |               comment               |
------------------------------------------------------------
|       vc2          |            2             | comment message for vc2 |
------------------------------------------------------------
============================================================
======
```

```
|                          VIRTUAL CLUSTER DATA NODE INFORMATION
|
=============================================================================
======
|NodeName|    IpAddress    |DistributionId| gnode |syncserver|DataState|
-----------------------------------------------------------------------
| node1    |172.168.83.13|          2           |FAILURE|                |              |
-----------------------------------------------------------------------
| node2    |172.168.83.14|          2           | OPEN    |     OPEN     |       0      |
-----------------------------------------------------------------------

2 data node
```

4.3.1.3 Virtual cluster management commands

4.3.1.3.1 Creating a virtual cluster

function

Generate a virtual cluster using the data nodes specified in the configuration file.

be careful

- To create a virtual cluster, users need to use the dbaUser user filled in the demo. options section when installing the cluster;
- The gcadmin createVC command can only create one VC at a time, meaning there can only be one information to create a VC in the configuration file.

grammar

```
gcadmin createvc <create_ vc.xml    |    e    example_ file_ name>
```

Table -419 Parameter Description

Parameter Name	explain
create_ vc.xml	The configuration file used to generate the VC, including the IP addresses and optional parameter VC name and comment information of all data nodes that generate the specified VC.
e	Generate a configuration demonstration file for creating a VC, which will not generate a VC when using this parameter.
example_ file_ name	The name of the generated rhetorical question when creating a

Parameter Name	explain
	VC configuration.

explain

- The installation script will automatically generate a configuration file gcChangeInfo.xml containing all data node IP information, which users can modify according to their own needs.

Example

Example 1: Generate a configuration demonstration file for creating VC.

```
$ gcadmin createvc e create_ vc.xml
$ cat create_ vc.xml
<? xml version='1.0' encoding="utf-8"?>
<servers>

    <rack>
        <node ip="vc data node ip"/>
        <!--- ... -->
        <node ip="vc data node ip"/>
    </rack>

    <vc_ name name="virtual cluster name no more than 64 bytes"/>
    <comment message="comment message no more than 60 bytes"/>

</servers>
```

Table -420 Configuration File Label Parameter Description

Parameter Name	explain
node ip	Generate the data node IP for the specified VC.
vc_ name	VC name, an optional parameter, if VC is not specified_ Name uses the system default name vcnamexxxxxx, which means vcname is incremented by 6 digits (starting from 1 and ending with 0 if less than 6 digits). The maximum length of vcname is 64 bytes. vcname must be an English character, with an underscore of '_' or a number. It does not support full width characters and the first letter must be an English character.
comment	Note information is an optional parameter. The maximum length of the comment is 60 bytes, only supports English letters, underscores "_", numbers, spaces "", commas "," periods "." and exclamation marks "!", and does not support full width characters.

Example 2: An example of modifying a configuration file is as follows:

```
<? xml version="1.0" encoding="utf-8"?>
<servers>
 <rack>
  <node ip="192.168.0.1"/>
  <node ip="192.168.0.2"/>
  <node ip="192.168.0.3"/>
 </rack>
 <comment message="comment message is optional parameter"/>
 <vc_ name name="vc_name is optional parameter">
</servers>
```

4.3.1.3.2 Delete virtual cluster

function

Remove the specified VC from the cluster.

be careful

- Before deleting a virtual cluster, it is necessary to first delete all user database tables and other database objects under VC, delete mirror relationships, and delete topology information.
- Before deleting a VC, it is necessary to delete the resource management information of the corresponding table (gbase. consumer_group, gbase. consumer_group_user, gbase. resource_plan, gbase. resource_pool, gbase. resource_plan_directive, gbase. resource_config, gbase. cluster_resource_pool_usage_history, gbase. resource_pool_events, etc.) in the gbase library. Otherwise, the VC cannot be deleted;
- The gcadmin rmVC command can only delete one VC at a time;
- If the VC specified for deletion does not exist, an error will be reported during execution;
- The nodes in the deleted VC become FreeNodes in the cluster. If you need to completely remove these nodes from the cluster, you can refer to 4.3.1.2.2 gcadmin rmnodes to remove nodes.

grammar

```
gcadmin rmvc <vc_ name>
```

Table -421 Parameter Description

Parameter Name	explain
vc_ name	The name of the VC to be deleted

4.3.1.3.3 Import Virtual Cluster

function

Import all data nodes from the existing cluster into the current cluster, becoming a VC of the current cluster.

be careful

- The imported cluster must be GBase 8a MPP Cluster V95 or above;
- The version of the imported cluster and the imported cluster must be consistent;
- The gcadmin importvc command can only import one VC at a time;
- After importing VC, the distribution of the original cluster remains unchanged;
- VC import with multiple distributions is not supported. The imported VC can only contain one distribution. If the VC contains multiple distributions before importing, all data must be redistributed according to the specified distribution before importing;
- After importing, all data nodes under the imported cluster will become a VC of the current cluster, and all management nodes will not be imported.

grammar

```
gcadmin importvc <import_ vc.xml | e example_ file_ name>
```

Table -422 Parameter Description

Parameter Name	explain
import_ vc.xml	Import the configuration file used by VC.
e	Generate a configuration demonstration file for importing VC. When using this parameter, VC will not be imported.
example_ file_ name	Generate the name of the configuration demonstration file for importing VC.

Example

Example 1: Generate a configuration demonstration file for creating VC.

```
$ gcadmin importvc e import_ vc.xml
gcadmin importvc vc ...
gcadmin importvc vc generate example file:[import_vc.xml] successful
$ cat import_ vc.xml
```

```
<? xml version='1.0' encoding="utf-8"?>
<import_ vc_ parameter>

    <source_ vc_ name name="old vc name to be imported"/>
    <target_ vc_ name name="new vc name after import to other cluster, could be same as source_vc_name"/>

    <imported_ vc_ gcluster_ ip ip_ list="the imported vc gcluster ip list, splitting by comma"/>

    <imported_ vc_ os_ dba_ user_ name os_ user_ name="the imported cluster os dba user name"/>
    <imported_ vc_ os_ dba_ password os_ password="the imported cluster os dba user password ciphertext"/>

    <imported_ vc_ db_ user_ name db_ user_ name="the imported vc database user name"/>
    <imported_ vc_ db_ password db_ password="the imported vc database password ciphertext, write " if no password"/>

    <importing_ os_ dba_ user_ name os_ user_ name="the importing cluster os dba user name"/>
    <importing_ os_ dba_ password os_ password="the importing cluster os dba user password ciphertext"/>

    <importing_ db_ user_ name db_ user_ name="the importing vc database user name"/>
    <importing_ db_ password db_ password="the importing vc database password ciphertext, write " if no password"/>

    <import_ vc_ timeout timeout="import vc timeout, unit is minute"/>

    <import_ vc_ include_ large_ data_ info include_ large_ data_ info="if import audit.log and resource_history when import vc , 0--not include 1--include"/>

</import_ vc_ parameter>
```

Table -423 Configuration File Label Parameter Description

Parameter Name	explain
source_ vc_ name	The VC name of the imported cluster.
target_ vc_ name	The new vcname after importing the cluster can use the original vcname
imported_ vc_ gcluster_ ip	List of gcluster node IPs imported into the cluster, separated by commas
imported_ vc_ os_	The dba username of the operating system being imported into

Parameter Name	explain
dba_ user_ name	the cluster
imported_ vc_ os_ dba_ password	The password of the dba user of the operating system imported into the cluster
imported_ vc_ db_ user_ name	The username of the database being imported into the cluster
imported_ vc_ db_ password	Database password imported into the cluster
importing_ os_ dba_ user_ name	The operating system dba username of the cluster performing the import operation
importing_ os_ dba_ password	The operating system dba user password of the cluster performing the import operation
importing_ db_ user_ name	The database username of the cluster performing the import operation
importing_ db_ password	The database user password of the cluster performing the import operation
import_ vc_ timeout	The timeout period for importing the cluster.
import_ vc_ include_ large_ data_ info	When the value is 1, import the audit logs and other information of the cluster containing the cluster; When the value is 0, importing the cluster does not include information such as audit logs for the cluster

4.3.1.3.4 Start virtual cluster

function

Start multiple VCs simultaneously.

explain

- Starting VC only starts the data node.

grammar

```
gcadmin startvc <vc_ name1 vc_ name2 ...> < os_ dba_ user_ name> <os_ dba_ password>
```

Table -424 Parameter Description

Parameter Name	explain
vc_ name1 vc_	The name of the VC to start.

Parameter Name	explain
name2 ...	
os_ dba_ user_ name	The operating system dbaUser name, which is the dbaUser filled in in demo. options.
os_ dba_ password	The password of the operating system dba user, which is the password of dbaUser filled in demo.options.

4.3.1.3.5 Stop virtual cluster

function

Stop multiple VCs simultaneously.

 **explain**

- Stopping VC only stops the data node.

grammar

```
gcadmin stopvc <vc_ name1 vc_ name2 ...> < os_ dba_ user_ name> <os_ dba_ password>
```

Table -425 Parameter Description

Parameter Name	explain
vc_ name1 vc_ name2 ...	The VC name to stop.
os_ dba_ user_ name	The operating system dbaUser name, which is the dbaUser filled in in demo. options.
os_ dba_ password	The password of the operating system dba user, which is the password of dbaUser filled in demo.options.

4.3.1.3.6 Rename virtual cluster

function

Change the name of an existing VC.

grammar

```
gcadmin renamevc <old_ vc_ name> <new_ vc_ name>
```

Table -426 Parameter Description

Parameter Name	explain
old_ vc_ name	The VC name before modification.
new_ vc_ name	The modified VC name.

4.3.1.3.7 View virtual clusters

function

View the name of the VC.

grammar

```
gcadmin
gcadmin showcluster vc <vcname>
$ gccli -e 'show vcs';
```

4.3.1.4 State management commands

4.3.1.4.1 Rmfeventlog command

function

Before performing a node replacement operation, use this command to delete all fevent logs (DDL fevent log, DML fevent log, DMLStorage fevent log) of the replaced node (which has been set to unavailable state).

 **warning**

- Use with caution, do not use non node replacement operations;
- After execution, the administrator will be prompted to confirm again.

grammar

```
gcadmin rmfeventlog ip
```

4.3.1.5 Information viewing command

4.3.1.5.1 Showdlevent command

function

View the log of data recovery information generated by DDL operations in the cluster.

The relevant commands refer to the following:

1. Gcadmin showdlevent: View the data recovery information generated by all DDL operations in the cluster.
2. Gcadmin showddlevent tablename semame nodeip: View the data recovery information generated by a partition of a table on a specific node due to DDL operations.
3. Gcadmin showddlevent tablename nodeip: View the data recovery information generated by a table on a node due to DDL operations.

be careful

If there is a DDL event in the system, it cannot be upgraded.

grammar

```
gcadmin showddlevent [<[[vc_name.]database_ name.] table_ name segname nodeip> | <[[vc_name.]database_ name.tablename nodeip> | <max_ fevent_ num>] [f] [vc vc_name]
```

Table -427 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name.
database_ name	Database name.
table_ name	Table name.
segname	The name of the table shard, such as creating a distribution table named t, where the table shard name on the first node is n1, the table shard name on the second node is n2,..., and so on.
nodeip	The IP address of the data node machine.
max_ fevent_ num	Gcadmin returns the dlevent log information based on the maximum number of entries set by the user. If the maximum

Parameter Name	explain
	number of entries set is greater than the total amount of existing information, all information is returned. Otherwise, a specified amount of information is returned. If the maximum number of entries is not specified, 16 entries will be returned by default, and subsequent feven logs will not be displayed.
f	Display query information in XML format.
vc vc_ name	Specify the name of the VC to display ddlevent.

Example

```
$ gcadmin showddlevent
Event count:2
Event ID:       2
ObjectName: test.t1
Fail Node Copy:
-----------------------------------------------------
NodeID: 2123999424    NodeIP:192.168.153.126    FAILURE

Fail Data Copy:
-----------------------------------------------------
NodeIP: 192.168.153.126 FAILURE



Event ID:       3
ObjectName: test.t2
Fail Node Copy:
-----------------------------------------------------
NodeID: 2123999424    NodeIP:192.168.153.126    FAILURE

Fail Data Copy:
-----------------------------------------------------
NodeIP: 192.168.153.126 FAILURE
```

4.3.1.5.2 **Showdmlevent command**

function

View the data recovery information of the cluster caused by DML operations.

The relevant reference commands are as follows:

1. Gcadmin showdmevent: View the data recovery information generated by the cluster due to DML operations.
2. Gcadmin showdmevent tablename semame nodeip: View the data recovery information generated by a partition of a table on a node due to DML operations.

be careful

If there is a feventlog in the system, it cannot be upgraded.

grammar

```
gcadmin showdmlevent [<[[vc_name.]database_ name.] tablename segname nodeip> | <max_ fevent_ num>] [f] [vc vc_name
```

Table -428 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name.
database_ name	Database name.
table_ name	Table name.
segname	The name of the table shard (view the shard name through showdistribution), such as creating a distribution table named t, where the table shard name on the first node is n1, the table shard name on the second node is n2,..., and so on.
nodeip	The IP of the node machine.
max_ fevent_ num	The gcadmin tool returns dmevent log information based on the maximum number of entries set by the user. If the maximum number of entries set is greater than the total amount of existing information, all information is returned. Otherwise, a specified amount of information is returned. If the maximum number of entries is not specified, 16 entries will be returned by default, and the subsequent feventlog will not be displayed.

Example

```
$ gcadmin showdmlevent
Event count:2
Event ID:       3
ObjectName: test.t1

Fail Data Copy:
------------------------------------------------------------
SegName: n3 SCN: 0    NodeIP: 192.168.153.126 FAILURE
SegName: n4 SCN: 0    NodeIP: 192.168.153.126 FAILURE



Event ID:       2
ObjectName: test.t4

Fail Data Copy:
------------------------------------------------------------
SegName: n4 SCN: 0    NodeIP: 192.168.153.126 FAILURE
SegName: n3 SCN: 0    NodeIP: 192.168.153.126 FAILURE
```

4.3.1.5.3 Showdmlstorageevent command

function

This command is used to display the fragmented complete recovery information of the tables in the current cluster.

be careful

If there is a DML storage event in the system, it cannot be upgraded.

grammar

```
gcadmin showdmlstorageevent [table_ID segname nodeip] | <max_ fevent_ num>]
[f] [vc vc_name]
```

Table -429 Parameter Description

Parameter Name	explain
table_ ID	The number of the table. You can use the system table information_ Query and obtain the schema. tables table.

Parameter Name	explain
segname	The name of the table shard (view the shard name through showdistribution), such as creating a distribution table named t, where the table shard name on the first node is n1, the table shard name on the second node is n2,..., and so on.
nodeip	The IP of the node machine.
max_ fevent_ num	The gcadmin tool returns dmlstorageevent log information based on the maximum number of entries set by the user. If the maximum number of entries set is greater than the total amount of existing information, all information is returned. Otherwise, a specified amount of information is returned. If the maximum number of entries is not specified, 16 entries will be returned by default, and subsequent dmlstorageevent logs will not be displayed.
f	Display cluster node information in XML format.
vc vcname	Specify the name of the virtual cluster where you want to view dmlstorageevent information.

Example

```
$ gcadmin showdmlstorageevent
Event count:2
Event ID:       5
ObjectName: test.t1
TableID: 26

Fail Data Copy:
-----------------------------------------------------
SegName: n2 NodeIP: 192.168.153.129 FAILURE



Event ID:       6
ObjectName: test.t2
TableID: 32

Fail Data Copy:
```

```
-----------------------------------------------------
SegName: n2 NodeIP: 192.168.153.129 FAILURE
```

4.3.1.5.4 Feventlog displays detailed information

- Grammar:

gcadmin showddlevent [detail]

gcadmin showdmlevent [detail]

gcadmin showdmlstorageevent [detail]

- explain:

The display syntax of feventlog includes the optional parameter 'detail'. Adding the 'detail' parameter will add four detailed information items on top of the original display information:

Time: The time generated by the feventlog, accurate to milliseconds;

Source: feventlog source (which node records it), displaying IP information;

SessionId: The sessionId of the source SQL that generates the feventlog. When the sessionId is empty, it is displayed as 0;

Cmd: The type of source SQL that generates the feventlog.

SQL type:

1) For the feventlog recorded by gcluster, the SQL type is displayed as follows:

INSERT、DELETE、UPDATE、LOAD、CREATE_ USER、CREATE_ DB、CREATE_ TABLE 、 CREATE_ VIEW 、 CREATE_ INDEX 、 CREATE_ PROCEDURE、CREATE_ FUNCTION、CREATE_ EVENT、RENAME_ USER、ALTER_ DB、ALTER_ TABLE、ALTER_ PROCEDURE、ALTER_ FUNCTION、ALTER_ EVENT、DROP_ USER、DROP_ DB、DROP_ TABLE、DROP_ VIEW、DROP_ INDEX、DROP_ PROCEDURE、DROP_ FUNCTION、DROP_ EVENT、TRUNCATE、GRANT、REVOKE、SELECT、CREATE_ SYNONYM、DROP_ SYNONYM、REBALANCE、CREATE_ DBLINK、DROP_ DBLINK、ALTER_ TABLESPACE、RESOURCE_ MANAGER。

Other SQL types are displayed as UNKNOWN.

2) For the feventlog of gcrecover records, the SQL type is displayed as GCRecover

3) For the feventlog recorded in the set command, the SQL type is displayed as

SET_ OPTION

For example, set self gcluster_ node_ status_ list="vcid:db.tb:segment:nodeid:datastate:scn"

- Example:

$ gcadmin showddlevent detail

Vc event count:1

Event ID: 2

ObjectName: testdb.test

Fail Node Copy:

Fail Data Copy:

SegName: n1 NodeIP: 192.168.146.25 FAILURE

Time:2023-05-06 11:48:26.630 Source:192.168.146.24 SessionId:358 Cmd:CREATE_ TABLE

SegName: n2 NodeIP: 192.168.146.25 FAILURE

Time:2023-05-06 11:48:26.630 Source:192.168.146.24 SessionId:358 Cmd:CREATE_ TABLE

$ gcadmin showdmlevent detail

Vc event count:1

Event ID: 2

ObjectName: testdb.test

Fail Data Copy:

SegName: n1 SCN: 3079 NodeIP: 192.168.146.25 FAILURE

Time:2023-05-06 11:48:36.587 Source:192.168.146.24 SessionId:358
Cmd:INSERT

4.3.1.5.5 Showcluster command

function

Display cluster node information.

be careful

- When half or more of the coordinator nodes in the cluster are offline, executing the gcadmin command will get stuck and not return. If any coordinator nodes in the cluster are found to be offline, please confirm the relevant information and repair the fault type in a timely manner;
- Only one parameter c and VC vcname can be entered, and entering them simultaneously will result in an error.

grammar

```
gcadmin showcluster [c | vc vcname] [d] [f]
```

Table 430 Parameter Description

Parameter Name	explain
c	Only display cluster coordinator node information.
vc vcname	Specify the name of the virtual cluster where you want to view information.
d	Display only cluster data node information.
f	Display cluster node information in XML format.

Example

Example 1: Display all node information.

```
$ gcadmin showcluster
CLUSTER STATE:              ACTIVE

=============================================================
|                    GBASE COORDINATOR CLUSTER INFORMATION  |
=============================================================
|    NodeName    |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
```

```
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
=================================================
|          GBASE VIRTUAL CLUSTER INFORMATION           |
=================================================
|       VcName       | DistributionId |     comment     |
-------------------------------------------------
|         vc1          |             1             | vc1comments |
-------------------------------------------------
|         vc2          |             2             | vc2comments |
-------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node
```

Example 2: Display all node information of VC1.

```
$ gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

=================================================
|          GBASE VIRTUAL CLUSTER INFORMATION           |
=================================================
|       VcName       | DistributionId |     comment     |
-------------------------------------------------
|         vc1          |             1             | vc1comments |
---------------------------------
=============================================================
=======
|                              VIRTUAL  CLUSTER  DATA  NODE  INFORMATION
|
=============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-------------------------------------------------------------------
| node1    |172.168.83.11|           1            |OPEN |     OPEN     |       0       |
-------------------------------------------------------------------
| node2    |172.168.83.12|           1            |OPEN |     OPEN     |       0       |
-------------------------------------------------------------------

2 data node
```

Example 3: Display only coordinator node information.

```
$ gcadmin showcluster c
CLUSTER STATE:                ACTIVE
CLUSTER MODE:                 NORMAL
```

```
============================================================
======
|                          GBASE  COORDINATOR  CLUSTER  INFORMATION
|
============================================================
======
|     NodeName     |     IpAddress     | gcware | gcluster | DataState |
------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
------------------------------------------------------------

3 coordinator node
```

Example 4: Display the data node information of vc2.

```
$ gcadmin showcluster vc vc2 d
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

================================================
|          GBASE VIRTUAL CLUSTER INFORMATION          |
================================================
|       VcName       | DistributionId |      comment      |
------------------------------------------------
|         vc2          |             2             | vc2comments |
------------------------------------------------
============================================================
=======
|                                VIRTUAL  CLUSTER  DATA  NODE  INFORMATION
|
============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
------------------------------------------------------------------
| node1    |172.168.83.13|           2            |OPEN |      OPEN     |       0       |
------------------------------------------------------------------
| node2    |172.168.83.14|           2            |OPEN |      OPEN     |       0       |
------------------------------------------------------------------

2 data node
```

Example 5: Display data node information in XML format.

```
$ gcadmin showcluster vc vc2 d f
<? xml version='1.0' encoding="utf-8"?>
<ClusterInfo>
        <CoordinatorClusterState>ACTIVE</CoordinatorClusterState>
        <VirtualClusterMode>NORMAL</VirtualClusterMode>
```

```
    <VirtualClusters>
        <VirtualCluster>
            <VcName>vc2</VcName>
            <DistributionId>2</DistributionId>
            <comment>vc2comments</comment>
        </VirtualCluster>
    </VirtualClusters>
    <VirtualClusterDataNodes>
        <DataServerNode>
            <NodeName>node1</NodeName>
            <IpAddress>172.168.83.13</IpAddress>
            <DistributionId>2</DistributionId>
            <gnode>OPEN</gnode>
            <syncserver>OPEN</syncserver>
            <DataState>0</DataState>
        </DataServerNode>
        <DataServerNode>
            <NodeName>node2</NodeName>
            <IpAddress>172.168.83.14</IpAddress>
            <DistributionId>2</DistributionId>
            <gnode>OPEN</gnode>
            <syncserver>OPEN</syncserver>
            <DataState>0</DataState>
        </DataServerNode>
    </VirtualClusterDataNodes>
    <DataNodeNumber>2</DataNodeNumber>
</ClusterInfo>
```

4.3.1.5.6 **Showfailover command**

function

Display all failover information currently retained in gcware.

Note: When a transaction is opened and contains multiple DMLs, it corresponds to a failover, but the displayed content will contain multiple DMLs.

Table -431 Display Column Description

Column Name	explain
commit id	The unique identifier of the failover, which is a 64 bit number.
database	Database name.
table	Table name.
scn	SCN number.
type	ddl/dml/rebalance。
create time	The time when the current node created the failover

Column Name	explain
	information.
state	The corresponding states for failover are as follows: ● Init: initialization, corresponding to the number 0 displayed ● add_ Res: Add cluster lock, corresponding to the number 1 displayed ● set_ Info: Set the failover information to display the corresponding number 2 ● set_ Status: Set the sharding status, corresponding to the number 3 displayed ● set_ rebalance_ Info: Set Rebalance information to display the corresponding number 4 ● set_ rebalance_ Status: Set the Rebalance status, corresponding to the number 5 displayed
original node	Originating node.
takeover node	The current takeover node is displayed as 0.0.0.0 if no takeover has occurred.
takeover number	The number of takeover attempts for failover, which is increased by 1 after GCware notifies GCluster to take over.

grammar

```
gcadmin showfailover [f]
```

Table -432 Parameter Description

Parameter Name	explain
f	Optional parameters, displaying information in XML format.

Example

```
$ gcadmin showfailover

+=============================================================
=============================================================
==========================+

|

GCLUSTER                                                FAILOVER

|

+=============================================================
=============================================================
```

```
============================+

+------------+------------+-----------+------------+-----------+-----------------+-------+---
-------------+-----------------+-----------------+

| commit id | database | table | scn | type | create
time | state | original node | takeover node | takeover number |

+------------+------------+-----------+------------+-----------+-----------------+-------+---
-------------+-----------------+-----------------+

| 1 | test | t1 | 1 | ddl |
20161019101114 | 5 | 192.168.153.130 | 0.0.0.0 | 0
|

+------------+------------+-----------+------------+-----------+-----------------+-------+---
-------------+-----------------+-----------------+
```

4.3.1.5.7 **Showfailover detail command**

function

Display the failover details for the specified commit.

Table -433 Description of Names

name	explain
failover_ information	Failover related information, including commit_ id, database, table, scn, type, create_ time, state, original_ node, takeover_ node, takeover_ number。 Please refer to Table 4-24 for detailed information.
content	Failover complete information, up to 256k.
status	The object state of the fail over operation, which corresponds to the state of which node and which partition. For example, node1.n1 init means that the n1 shard on node1 node has not yet been submitted and is in an initialization state.
rebalance_ information	Rebalance unique information (including distribution_id, current_scn, current_step, and middle table name), where ddl dml is displayed as an empty label.

name	explain
sdm	Rebalance unique information, ddl dml is displayed as an empty label. Contains the following fields: ● NodeId. Suffix: A shard of a node; ● CurRowid: The row label of the row to which rebalance is executed; ● Blockid BlockNum: The row label of the row where the previous batch of rebalance was executed.

grammar

```
gcadmin showfailoverdetail <commitId> [xml_file_name]
```

Table -434 Parameter Description

Parameter Name	explain
commitid	The unique identifier of failover, which must be entered.
xml_ file_ name	The file name to save the failover information, optional parameter. If left blank, the failover information will be printed to the screen.

Example

```
$ gcadmin showfailoverdetail 1
<? xml version='1.0' encoding="utf-8"?>
<failover_ detail>
    <failover_ information>
        <commit_ id>1</commit_ id>
        <database>test</database>
        <table>t1</table>
        <scn>1</scn>
        <type>ddl</type>
        <create_ time>20161019101114</create_ time>
        <state>5</state>
        <original_ node>192.168.153.130</original_ node>
        <takeover_ node>0.0.0.0</takeover_ node>
        <takeover_ number>0</takeover_ number>
    </failover_ information>
    <content>create table t1(a int)</content>
    <status>
```

```
    </status>
    <rebalance_ information>
        <distribution_ id>1</distribution_ id>
        <current_ scn>10</current_ scn>
        <current_ step>3</current_ step>
        <table>tmpt1</table>
    </rebalance_ information>
    <sdm>
        <slice_ dm>from_ slice node1.n1.row10.block_ id1</slice_ dm>
        <slice_ dm>from_ slice node2.n2.row9.block_ id2</slice_ dm>
        <slice_ dm>from_ slice node3.n3.row8.block_ id3</slice_ dm>
     </sdm>
</failover_ detail>
```

4.3.1.5.8 **Showlock command**

function

View the locks currently present in the cluster.

This includes the name of the lock, the owner of the lock, the creation time of the lock, the note of the lock, whether the lock has been locked, and the type of lock.

Table -435 shows column descriptions

Column Name	explain
lock name	The name of the lock.
owner	The node IP that initiated the lock operation.
content	The note information of the lock.
create time	The creation time of the lock (based on the time of the locking node).
locked	Represents the status of the lock, i.e. whether it is held, queued, not queued, etc
type	The type of lock, where S represents a shared lock and E represents an exclusive lock.

grammar

```
gcadmin showlock [f]
```

Table 436 Parameter Description

Parameter Name	explain
f	Optional parameter, displaying lock information in XML format.

Example

```
$ gcadmin showlock
+=============================================================================
=======+
|                                                                GCLUSTER LOCK
|
+=============================================================================
=======+
+-------------+-------------+-------------+--------------+------+----+
|   Lock name   |      owner      |     content     | create time   |locked|type|
+-------------+-------------+-------------+--------------+------+----+
|gc-event-lock|172.168.83.11|global master|20200509093159| TRUE | E    |
+-------------+-------------+-------------+--------------+------+----+
|gc-event-lock|172.168.83.13|global master|20200629130253|FALSE | E    |
+-------------+-------------+-------------+--------------+------+----+
|gc-event-lock|172.168.83.12|global master|20200629130256|FALSE | E    |
+-------------+-------------+-------------+--------------+------+----+
Total : 3
```

Example 2:

```
$ gcadmin showlock f
<? xml version='1.0' encoding="utf-8"?>
<gcluster_ lock>
    <locks>
        <lock lock_ name="gc-event-lock" owner="172.168.83.11"
content="global master" create_ time="20200509093159" locked="TRUE"
type="E"/>
        <lock lock_ name="gc-event-lock" owner="172.168.83.13"
content="global master" create_ time="20200629130253" locked="FALSE"
type="E"/>
        <lock lock_ name="gc-event-lock" owner="172.168.83.12"
content="global master" create_ time="20200629130256" locked="FALSE"
type="E"/>
    </locks>
    <Total count="3"/>
</gcluster_ lock>
```

4.3.1.6 Other commands

4.3.1.6.1 Help command

function

View help information for all command parameters of gcadmin.

grammar

```
gcadmin --help
```

4.3.1.6.2 Version command

function

View the version information of gcadmin.

grammar

```
gcadmin <-V | --version>
```

4.3.2 VC management

4.3.2.1 Accessing VC

4.3.2.1.1 Set default VC

Operation scenario

After specifying the default VC for database users, they log in to the virtual cluster and directly enter the default VC.

grammar

```
set default_ vc for user_ name = vc_ name
```

Table 437 Parameter Description

Parameter Name	Description
user_ name	Specify the username to set the default VC

Parameter Name	Description
vc_ name	Specify the default virtual cluster name for the user

 **explain**

- If the user wants to cancel the default VC, they need to set VC_ The name is NULL.

4.3.2.1.2 Set permissions

Operation scenario

Only when a user has access permissions set to a certain VC can they access that VC.

grammar

There are two types of VC authorization libraries, tables, columns, functions, and stored procedures that DBA authorized users can access:

- Do not specify a virtual cluster name, that is, authorize the objects under the current VC to the USER. The authorization method is as follows:

```
grant all on *.* to user;
grant all on db.* to user;
grant all on db.table to user;
grant all on function db. func to user;
grant all on procedure db. proc to user;
```

- Specify the virtual cluster name, which authorizes the objects under the specified VC to the USER. The authorization method is as follows:

```
grant all on vc.*.* to user;
grant all on vc.db.* to user;
grant all on vc.db.table to user;
grant all on function vc.db.func to user;
grant all on procedure vc.db.proc to user;
```

 **explain**

For detailed instructions on permission settings, please refer to the section 4.9.1.3 Permission Management.

4.3.2.1.3 Revoke permissions

grammar

There are two ways to reclaim VC access permissions for users:

- Not specifying a virtual cluster name means revoking the object permissions of the USER under the current VC. The method is as follows:

```
revoke all on *.* from user;
revoke all on db.* from user;
revoke all on db.table from user;
```

- Specify the virtual cluster name, that is, revoke the object permissions of USER under the specified VC. The method is as follows:

```
revoke all on vc.*.* from user;
revoke all on vc.db.* from user;
revoke all on vc.db.table from user;
```

4.3.2.1.4 Switch VC

grammar

```
use vc vc_ name;
```

4.3.2.2 Operation under VC

4.3.2.2.1 DQL

Function Description

If you want to access a library table that is not the current VC, you need to explicitly specify the VC name. Accessing the library table of the current VC can omit the VC name.

Example

```
gbase> select * from vc1.db1.t1;
+------+------+
| a    | b    |
+------+------+
```

```
|      1 | test |
+------+------+
1 row in set (Elapsed: 00:00:00.02)
```

4.3.2.2.2 DML

Function Description

Support data migration between VC with access permissions.

Example

```
gbase> insert into vc2.db2.t2 select * from vc1.db1.t1;
Query OK, 1 row affected (Elapsed: 00:00:00.24)
Records: 1  Duplicates: 0  Warnings: 0
```

4.3.2.2.3 DDL

Function Description

Support DDL execution between VC with relevant permissions.

Example

Example: Taking the current VC as vc1 as an example.

```
gbase> use vc vc1;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)

gbase> create database vc2.db3;
Query OK, 1 row affected (Elapsed: 00:00:00.02)
```

4.3.3 Virtual cluster image

4.3.3.1 Function Description

The virtual cluster mirroring function is a function based on the concept of virtual cluster multi VC. It is mainly used to set the mirroring relationship between the tables of the virtual cluster between two VCs, so that users can synchronize their write operations (DDL, DML, LOAD) on any table data in real-time to the other table corresponding to the mirroring relationship. Mirrors are mutual, equal, and do not have a primary or secondary relationship.

This feature requires tables belonging to two different VCs to have the same database name, table name, table structure, and table distribution in order to have a mirror relationship. After deleting an existing mirror relationship, both tables on both sides of the mirror relationship are available.

Note: If an image is created for an existing table, it is necessary to give the operating user permission to the image table at the same time, so that the user's operations can be synchronized to the image table.

4.3.3.2 Command Description

The commands related to the virtual cluster image function mainly include creating an image and deleting an image. To use the mirroring function, it is necessary to ensure that the two VCs where the main table and mirror table are located have the same number of main partitions and the HASHMAP is the same.

4.3.3.2.1 Comparing the similarities and differences of hashmaps between VC's

The VC has been initialized and the HASHMAP of the two VCs can be confirmed to be the same using the following method:

1. View the distribution of VC.

```
gcadmin showdistribution vc    vc1
```

The mirroring function requires the same number of primary partitions for the distribution of two VCs. For example, if VC1's distribution has three main shards, VC2's distribution also requires three main shards, and there is no requirement for the number of backup shards.

2. Compare the HASHMAP of two VCs to see if they are the same. If the results of executing the following SQL are consistent, as shown in the example below, it indicates that the HASHMAP1 of VC1 is the same as the HASHMAP2 of VC2.

```
gbase> select distinct(res.r), count(res.r) from (select count(*) r    from
gbase.nodedatamap t where t.data_ distribution_ id in (1,2) group by
t.hashkey, t.nodeid) res group by res.r;
+---+--------------+
| r | count(res.r) |
+---+--------------+
| 2 |        65536 |
+---+--------------+
```

4.3.3.2.2 VC assimilation hashmap

Assimilate HASHMAP: Create the same hashmap for the two VCs that the mirror table

belongs to:

1. Create the same distribution for two clusters;
2. Create the same HASHMAP during initialization:

   Grammar:

```
INITNODEDATAMAP FROM VC1;
```

   Example:

```
INITNODEDATAMAP FROM VC1;
```

For example, if there are two VCs, VC1 has already initialized HASHMAP using the INITNODEDATAMAP command, VC2 can use INITNODEDATAMAP from VC1 to initialize HASHMAP, so that the HASHMAP of VC1 and VC2 will be the same.

4.3.3.2.3 create mirror

The virtual cluster mirroring function provides three ways to create table images. namely:

1. Create a single table image;
2. Create table images on a library basis;
3. Create both primary and mirror tables simultaneously;
4. Create the default image VC for the library.

- Create a single mirror table

Create Mirror Table Command:

```
ALTER TABLE VC1.DB.T1 CREATE MIRROR TO VC2;
```

The above command is used for scenarios where the VC1. DB.T1 table and VC2. DB library already exist, but the VC2. DB.T1 table does not exist. Create a T1 table, synchronize data, and create a mirroring relationship under VC2. DB, so that VC1. DB.T1 is exactly the same as VC2. DB.T1, and synchronize any side of the mirroring relationship table to the other side in subsequent operations. If the VC2.DB.T1 table exists, an error will be reported.

Force Mirror Creation Command:

```
ALTER TABLE VC1.DB.T1 CREATE MIRROR TO VC2 FORCE;
```

The above command uses the FORCE parameter for scenarios where the VC2. DB.T1 table exists. It will force the creation of a mirror table for VC1. DB.T1 in VC2. DB, making VC2. DB.T1 exactly the same as VC1. DB.T1, and synchronizing any side of the mirror relationship table to the other side in subsequent operations.

be careful

When using the forced creation mirror table SQL, when the table structure of VC1. DB.T1 and VC2. DB.T1 is different, this command will delete the VC2. DB.T1 table, then use the table creation SQL of VC1. DB.T1 table to recreate the VC2. DB.T1 table, and finally synchronize the VC1. DB.T1 table data to VC2. DB.T1 table.

Tables that have already established an image relationship can be directly deleted, and once the table is deleted, the image relationship will also be deleted.

- Create a table image on a library basis

```
ALTER DATABASE VC1.DB CREATE MIRROR TO VC2;
```

It is used in the scenario where the VC1.DB library and VC2.DB library both exist. In VC2.DB, create all non identical tables and non identical stored procedures and functions in VC1.DB. At the same time, synchronize data and create an image relationship for these newly created non identical tables, so that the structure and data of the two tables that created the image relationship are completely consistent (stored procedures and functions have no image relationship). Any subsequent changes made to the tables on either side of the mirroring relationship will be synchronized to the tables on the other side of the mirroring relationship, making the two tables completely consistent.

If there is a table or stored procedure or function with the same name of VC1.DB in VC2.DB, the entire sql will return the creation failure information of the table, stored procedure and function with the same name in the way of warning.

Tables in VC1. DB	Tables in VC2. DB	Execution results of this function
T		Create a mirror table for T in VC2. DB
T2	T2	There is no change in T2 in VC2. DB, while warnings report that T2 already exists
	T3	There is no change in T3 in VC2. DB and there is no impact
procedure1		Create procedure1 stored procedure in VC2. DB, but there is no mirroring relationship
	procedure2	Procedure2 in VC2. DB is not affected and remains unchanged
When creating an image relationship, only the image table of VC1.DB will be created in VC2.DB, and the tables, stored procedures, and functions in VC1.DB will not change. After the		

Tables in VC1. DB	Tables in VC2. DB	Execution results of this function
creation of the mirror relationship is successful, there is no primary or secondary distinction between the tables on both sides of the mirror relationship. Operations on either side will be mapped to the tables on the other side of the relationship, maintaining complete consistency between the tables on both sides.		

This function is mainly used to batch create images for tables under the library, that is, to create images for multiple tables simultaneously, with the concurrency number determined by the parameter gcluster_ mirror_ parallel_ Count specified.

 **be careful**

gcluster_ mirror_ parallel_ Count is a session level parameter. It is necessary to set the parallelism before execution, otherwise changing this parameter during the process of creating the mirror table will not take effect.

After successful creation, it can be found in gbase.table_ View the mirror relationship of tables in distribution:

select vc_ id,index_ name,mirror_ vc_ id from gbase.table_ distribution where dbname='***';

Command to force the creation of a library level mirror table (using the Force parameter):

```
ALTER DATABASE VC1.DB CREATE MIRROR TO VC2 FORCE;
```

It is used in the scenario where both VC1.DB and VC2.DB databases exist. Create all the tables, stored procedures and functions in VC1.DB in VC2.DB. At the same time, synchronize the data of these newly created tables and create an image relationship, so that the two tables that have created an image relationship are completely consistent. Any subsequent changes made to the tables on either side of the mirroring relationship will be synchronized to the tables on the other side of the mirroring relationship, making the two tables completely consistent.

Tables in VC1. DB	Tables in VC2. DB	Execution results of this function
T		Create a mirror table for T in VC2. DB
T2	T2	Delete T2 in VC2. DB, and create a mirror table for VC1. DB. T in VC2. DB
	T3	There is no change in T3 in VC2. DB and there is no impact
procedure1		Create procedure1 stored procedure in VC2. DB, but there is no mirroring relationship
	procedure2	Procedure2 in VC2. DB is not affected and

Tables in VC1. DB	Tables in VC2. DB	Execution results of this function
		remains unchanged

 **be careful**

The difference between creating a table image based on a library with and without force is the handling of tables with the same name:

SQL without force does not create mirror relationships for tables with the same name, and warnings will report a failure message;

Adding force to SQL to force the creation of a mirror relationship on a table with the same name will delete the table with the same name in VC2, and rebuild the table with the same name in VC1 in VC2, so that the table structure and data in VC2 are consistent with those in VC1.

- Create both primary and mirror tables simultaneously

  Add the MIRROR TO parameter to the CREATE TABLE statement to create both the primary and mirror tables simultaneously.

```
CREATE TABLE VC1.DB.T1(A INT, B VARCHAR(10)) MIRROR TO VC2;
CREATE TABLE VC1.DB.T1 MIRROR TO VC2 AS SELECT * FROM
VC1.DB.T
```

  If there is a T1 table in VC2. DB, an error will be reported.

```
gbase> create table vc1.mirrdb.tx mirror to vc2 as select * from vc2.mirrdb.x;
Query OK, 1 row affected (Elapsed: 00:00:05.06)
```

 **be careful**

The syntax does not support the force keyword

Creating table like specifying mirror tables is not supported

- Create a default image VC for the library

  Set the VC name where the default image of the VC1. DB library is located (hereinafter referred to as the default image VC), for scenarios where VC1. DB and VC2. DB already exist. If VC2. DB does not exist, an error will be reported.

```
ALTER DATABASE VC1.DB SET DEFAULT MIRROR = VC2;
```

When setting the default image VC, it has no impact on the existing tables, stored procedures, and views in the VC1. DB library and VC2. DB library.

After setting the default image VC, the previously existing tables, stored procedures, and views in VC1. DB and VC2. DB libraries remain unchanged, and there is no mirroring relationship. Any operation (including deleting and rebuilding objects with the same name) will not perform mirror synchronization.

After setting the default image VC, the mirroring relationship between VC1. DB library and VC2. DB library is successfully established. The newly created tables, views, stored procedures, and operations on these objects on either side of the mirroring relationship (VC1. DB or VC2. DB) will be synchronized to the other side of the mirroring relationship at the same time, keeping the newly created tables, views, and stored procedures on both sides of the mirroring relationship consistent.

Tables in VC1. DB	Tables in VC2. DB	Execution results of this function
Existing objects before setting default VC		
T、procedure1 DDL、DML	T3、procedure2 DDL、DML	Out of sync, no changes, no mutual impact
T2 DDL、DML	T2 DDL、DML	Out of sync, no changes, no mutual impact
Object to operate on after setting default VC		
Table 、 view 、 procedure DDL、DML		Synchronize to VC2. DB
	Table 、 view 、 procedure DDL、DML	Synchronize to VC1. DB

Delete the default image VC of the library using the following SQL:

```
ALTER DATABASE VC1.DB SET DEFAULT MIRROR = null;
```

4.3.3.2.4 **delete mirror**

- Delete a mirror of a single table

  The mirror command to delete a table is as follows:

  ```
  ALTER TABLE VC1.DB.T1 DELETE MIRROR;

  perhaps

  ALTER DATABASE VC1.DB.T1 SET DEFAULT MIRROR = NULL;
  ```

  After deleting the mirror of the table, VC1. DB.T1 and VC2. DB.T1 no longer have a mirror relationship, but the two tables still exist. The write operation on one table will no longer be synchronized to the other table.

  When deleting the mirror of a table, it is required that both VC1. DB.T1 and VC2. DB.T1 tables are available, meaning that after the mirror relationship is broken, the two tables are still two independent and available tables.

- Delete table images on a library basis

The command is as follows:

```
ALTER DATABASE VC1.DB DELETE MIRROR;

perhaps

ALTER DATABASE VC1.DB SET DEFAULT MIRROR = NULL;
```

This command will delete the mirroring relationships of all tables under VC1. DB.

 **be careful**

- The delete mirror function only removes the mirror relationship of the table, without deleting the table and its data. The mirror table will be rebuilt at the cluster layer, causing the table's tid to be different. For example, VC1. DB.T1 and VC2. DB.T1 have a mirror relationship. After deleting the mirror, the two tables no longer have a mirror relationship, and their tids are no longer the same. However, the two tables and their data still exist;
- Delete either of the two tables containing mirror relationships, and the mirror relationship will also be deleted accordingly;
- Tables with mirror relationships in the library cannot be directly deleted. All mirror relationships in the library must be deleted before the library can be deleted.

4.3.3.2.5 Viewing Mirror Relationships

- To view the mirroring relationship of a single table:

  The mirror relationship of the table is stored in the table of the system library gbase_ In the distribution table, the following SQL can be used to view the image relationship:

```
select vc_ id,index_ name,mirror_ vc_ id from gbase.table_ distribution;
```

An example is as follows:

```
gbase> select vc_ id,index_ name,mirror_ vc_ id from gbase.table_
distribution where dbname='testdb' order by vc_ id;
    +---------+---------------+--------------+
    | vc_ id     | index_ name       | mirror_ vc_ id |
    +---------+---------------+--------------+
    | vc00001 | testdb.t          | NULL               |
    | vc00001 | testdb.t2         | vc00002          |
    | vc00001 | testdb.vc1new | NULL               |
    | vc00002 | testdb.t          | NULL               |
    | vc00002 | testdb.x          | NULL               |
```

```
| vc00002 | testdb.t2          | vc00001            |
+---------+--------------------+--------------------+
6 rows in set (Elapsed: 00:00:00.07)
```

- View the default image VC of the library

```
show mirror databases;
```

An example is as follows:

```
gbase> show mirror databases;
+--------------------+-----+-----------+
| Database           | VC  | MIRROR_ VC |
+--------------------+-----+-----------+
| information_ schema |     |           |
| performance_ schema |     |           |
| gbase              |     |           |
| gctmpdb            |     |           |
| gclusterdb         | vc2 |           |
| mirrdb             | vc2 |           |
| testdb             | vc2 | vc1       |
+--------------------+-----+-----------+
7 rows in set (Elapsed: 00:00:00.43)
```

4.3.3.2.6 Restrictions on the use of mirroring function

- High availability aspect

1. Failover is not supported during the execution of the following SQL statements. If the execution node is abnormal, it may cause residual files to exist. It is necessary to execute the relevant statements again or add a force to solve the problem

Create/delete a single table mirror: Alter table create/delete mirror

Create/delete table images per library: Alter database create/delete mirror

Create default mirror for library VC: Alter database set default mirror

3. When executing the following SQL, if the image VC is not available, the SQL will report an error and fail to exit

Create a single table image: Alter table create mirror

Create a table image based on a library: Alter database create mirror

Create both primary and mirror tables: CREATE TABLE MIRROR TO

CREATE TABLE MIRROR AS SELECT

4. If there is a table creation failure in Alter database create mirror, it will be prompted

in warnings, and SQL will exit without reporting an error

- In terms of data operation

1. dml and select operations for mirror tables:

When there are available shards for the current VC or mirror VC, this table is available

When performing multi table association operations, all related tables under the same distribution need to be available

4.3.3.3 Sample Description

- Build environment examples

1. Install the virtual cluster first.
2. Create two VCs, vc1 and vc2.
3. Create a distribution under vc1 and vc2 respectively.

```
gcadmin distribution gcChangeInfo.xml p 1 d 1 vc vc1
gcadmin distribution gcChangeInfo.xml p 1 d 1 vc vc2
```

4. Initialize the hashmap for vc1 and vc2.

```
gccli -uroot -Dvc1. - e"initnodedatamap"
gccli -uroot -Dvc2. - e"initnodedatamap from vc1"
```

- Sample Mirror Function

```
-- prepare data
drop database if exists vc1.mirror_ test;
drop database if exists vc2.mirror_ test;
create database vc1.mirror_ test;
create database vc2.mirror_ test;


create table vc1.mirror_ test.t_ rand
      (l_orderkey bigint, l_partkey bigint, l_suppkey bigint,
      l_ linenumber bigint, l_ quantity decimal(15,2),
      l_ extendedprice decimal(15,2), l_ discount decimal(15,2),
      l_ tax decimal(15,2), l_ returnflag char(1), l_ linestatus char(1),
      l_ shipdate date, l_ commitdate date, l_ receiptdate date,
      l_ shipinstruct char(25), l_ shipmode char(10), l_ comment varchar(50));
create table vc1.mirror_ test.t_ hash
```

```
        (l_orderkey bigint, l_partkey bigint, l_suppkey bigint,
        l_ linenumber bigint, l_ quantity decimal(15,2),
        l_ extendedprice decimal(15,2), l_ discount decimal(15,2),
        l_ tax decimal(15,2), l_ returnflag char(1), l_ linestatus char(1),
        l_ shipdate date, l_ commitdate date, l_ receiptdate date,
        l_ shipinstruct char(25), l_ shipmode char(10), l_ comment varchar(50))
        distributed by ('l_orderkey');

create table vc1.mirror_ test.t_ rep
        (l_orderkey bigint, l_partkey bigint, l_suppkey bigint,
        l_ linenumber bigint, l_ quantity decimal(15,2),
        l_ extendedprice decimal(15,2), l_ discount decimal(15,2),
        l_ tax decimal(15,2), l_ returnflag char(1), l_ linestatus char(1),
        l_ shipdate date, l_ commitdate date, l_ receiptdate date,
        l_ shipinstruct char(25), l_ shipmode char(10), l_ comment varchar(50))
        replicated;
insert into vc1.mirror_ test.t_ rand values

('1','310379','15395','1','17','23619.12','0.04','0.02','N','O','1996-03-13','1996-02-12','
1996-03-22','DELIVER IN PERSON','TRUCK','blithely regular ideas caj'),

('2','212340','2361','1','38','47588.54','0.00','0.05','N','O','1997-01-28','1997-01-14','1
997-02-02','TAKE BACK RETURN','RAIL','carefully ironic platelets against t'),

('3','8594','3595','1','45','67616.55','0.06','0.00','R','F','1994-02-02','1994-01-04','199
4-02-23','NONE','AIR','blithely s'),

('4','176070','11095','1','30','34382.10','0.03','0.08','N','O','1996-01-10','1995-12-14','
1996-01-18','DELIVER IN PERSON','REG AIR','special dependencies am'),

('5','217139','17140','1','15','15841.80','0.02','0.04','R','F','1994-10-31','1994-08-31','
1994-11-20','NONE','AIR','unusual, even instructio'),

('6','279271','4285','1','37','46259.62','0.08','0.03','A','F','1992-04-27','1992-05-15','1
992-05-02','TAKE BACK RETURN','TRUCK','ruthlessly unusual warhorses sleep
```

```
slyly af'),

('7','364104','19159','1','12','14017.08','0.07','0.03','N','O','1996-05-07','1996-03-13','
1996-06-03','TAKE BACK RETURN','FOB','pending requests sleep furiously
above'),

('32','165408','15425','1','28','41255.20','0.05','0.08','N','O','1995-10-23','1995-08-27'
,'1995-10-26','TAKE BACK RETURN','TRUCK','ironic requests dazzle. final d'),

('33','122671','17690','1','31','52503.77','0.09','0.04','A','F','1993-10-29','1993-12-19'
,'1993-11-08','COLLECT COD','TRUCK','slyly even requests are f'),

('34','176723','1732','1','13','23396.36','0.00','0.07','N','O','1998-10-23','1998-09-14','
1998-11-06','NONE','REG AIR','regular deposits grow. regu');
insert into vc1.mirror_ test.t_ hash select * from vc1.mirror_ test.t_ rand;
insert into vc1.mirror_ test.t_ rep select * from vc1.mirror_ test.t_ rand;

-- test create del mirror table
alter table vc1.mirror_ test.t_ rand create mirror to vc2;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
and index_ name='mirror_ test.t_ rand';
select * from vc1.mirror_ test.t_ rand where l_ orderkey not in (select l_orderkey
from vc2.mirror_test.t_rand);
alter table vc1.mirror_ test.t_ hash create mirror to vc2;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
and index_ name='mirror_ test.t_ hash';
select * from vc1.mirror_ test.t_ hash where l_ orderkey not in (select l_orderkey
from vc2.mirror_test.t_hash);
alter table vc1.mirror_ test.t_ rep create mirror to vc2;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
and index_ name='mirror_ test.t_ rep';
select * from vc1.mirror_ test.t_ rep where l_ orderkey not in (select l_orderkey
```

```
from vc2.mirror_test.t_rep);

alter table vc1.mirror_ test.t_ rand delete mirror;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
and index_ name='mirror_ test.t_ rand';
select * from vc1.mirror_ test.t_ rand where l_ orderkey not in (select l_orderkey
from vc2.mirror_test.t_rand);
alter table vc1.mirror_ test.t_ hash delete mirror;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
and index_ name='mirror_ test.t_ hash';
select * from vc1.mirror_ test.t_ hash where l_ orderkey not in (select l_orderkey
from vc2.mirror_test.t_hash);
alter table vc1.mirror_ test.t_ rep delete mirror;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
and index_ name='mirror_ test.t_ rep';
select * from vc1.mirror_ test.t_ rep where l_ orderkey not in (select l_orderkey
from vc2.mirror_test.t_rep);
alter table vc2.mirror_ test.t_ rand create mirror to vc1 force;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
and index_ name='mirror_ test.t_ rand';
select * from vc1.mirror_ test.t_ rand where l_ orderkey not in (select l_orderkey
from vc2.mirror_test.t_rand);
alter table vc2.mirror_ test.t_ hash create mirror to vc1 force;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
and index_ name='mirror_ test.t_ hash';
select * from vc1.mirror_ test.t_ hash where l_ orderkey not in (select l_orderkey
from vc2.mirror_test.t_hash);
alter table vc2.mirror_ test.t_ rep create mirror to vc1 force;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
```

```
and index_ name='mirror_ test.t_ rep';
select * from vc1.mirror_ test.t_ rep where l_ orderkey not in (select l_orderkey
from vc2.mirror_test.t_rep);

-- test create del database mirror
alter database vc1.mirror_ test create mirror to vc2;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
and dbname='mirror_ test' order by index_ name;
alter database vc1.mirror_ test delete mirror;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
and dbname='mirror_ test' order by index_ name;
alter database vc1.mirror_ test create mirror to vc2 force;
select vc_ id, index_ name, mirror_ vc_ id from gbase.table_ distribution where
vc_ id in (select ID from    information_schema.vc where name in ('vc1', 'vc2'))
and dbname='mirror_ test' order by index_ name;

drop database vc1.mirror_ test;
drop database vc2.mirror_ test;
```

4.4 Command Line Tools

4.4.1 gccli

summary

Gccli is a command-line database connection tool that comes with GBase 8a MPP Cluster. It can be installed independently on a machine in a non cluster environment and only supports the Linux operating system supported by GBase 8a MPP Cluster.

function

All legitimate SQL and SQL files can be executed through the gccli tool.

install files

Gccli installation package: gccli-9.5.2. xx OSversion platform. tar. bz2

explain

gccli_ install ----------gccli_ Install.sh (client package installer)
| |
---------- gccli_ Standalone. tar. bz2 (client package)

Tool installation

Step 1

Use the system user to decompress using the tar command in command line mode. The decompression command is as follows:

```
$ tar xjf gccli-9.5.3.17-redhat7.3-x86_ 64.tar.bz2
```

Step 2

Then copy the extracted folder gccli_ From the content in install to the installation path, execute the installation program in the installation path:

```
./gccli_ install.sh gccli_ standalone.tar.bz2
```

Step 3

The installation program executed successfully, and the screen displays as follows:

```
gcluster/
gcluster/config/
gcluster/config/gbase_ 8a_ gcluster.cnf
gcluster/server/
```

```
gcluster/server/lib/
gcluster/server/lib/gbase/
gcluster/server/lib/gbase/libgclusterclient_ r.so.16
gcluster/server/bin/
gcluster/server/bin/gbase
Installation finished.
Please use /home/gbase/gccli_ install/gcluster/server/bin/gccli
```

Users can access/home/gbase/gccli_ Install/gcluster/server/bin/gccli for cluster client use.

grammar

```
gccli –u<username> -p<password> [-h<ipaddress>] [-P<port>]
[-D<databasename>] [--nice_time_format] [-c] [-f] [-v[v][v]] [-e] [<]
```

Table -438 Parameter Description

Parameter Name	explain
–u<username>	Name of the user connecting to the database
-p<password>	User password for connecting to the database
-h<ipaddress>	Log in to the IP address of the cluster node, default to 127.0.0.1, optional parameter. If multiple IP addresses are specified, the high availability feature of gccli is enabled, with IP addresses separated by ',', such as 192.168.100.10192.168.100.11192.168.100.12
-P<port>	The port number used by the cluster, default 5258, optional parameter
-D<databasename>	Optional parameter - D parameter value, specifying the default database for login (database must exist), optional parameter
--nice_ time_ forma	Specify the minimum precision of user operation time, using this parameter, accurate to milliseconds, not used, accurate to seconds. Optional parameters.
-c	Use this parameter to use the hint optimization method. Optional parameters;
--force，-f	When batch executing SQL files, if there is an SQL execution error in the middle, force the subsequent SQL to be executed, optional parameters;
--verbose，-v	Verbose mode. Generate more output. You can use this opti on multiple times to generate more output. (For example, - v - v - v can even generate table output formats in batch mode).
--version，-V	Display version information and exit.

Parameter Name	explain
--vertical，-E	Vertically output the rows output by the query. Without this option, the vertical output of a single statement can be specified by ending with G.
--execute=statement, -e statement	Execute the statement and exit, which can be multiple statements separated by ';', with optional parameters;
--silent，-s	Silent mode. Generate less output. You can use this option multiple times to produce less output.
--skip-column-names，-N	Do not include column names in the results.
--skip-line-numbers，-L	Do not write line numbers in error messages. It is useful when you want to compare result files that include error messages.
--html，-H	Generate HTML output.
	The method by which EOF receives input can be used for batch execution of SQL files, with optional parameters.

Example

Example 1

```
[ gbase@gcluster1    etc]$ gccli -ugbase -p

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.    All Rights Reserved.

gbase> use vc myvc;
Query OK, 0 rows affected (Elapsed: 00:00:00.01)
```

Example 2

```
$ gccli -uroot –p****** -Dgbase    -vvv -e"select count(*) from user;select user from user";
--------------
select count(*) from user
--------------

+----------+
| count(*) |
+----------+
|              2 |
```

```
+----------+
1 row in set (Elapsed: 00:00:00.00)


--------------
select user from user
--------------


+------------------+
| user             |
+------------------+
| gbase            |
| root             |
+------------------+
2 rows in set (Elapsed: 00:00:00.00)


Bye
```

Example 3

```
$ cat 1.sql
select count(*) from user;
select user from user;
$ gccli -uroot –p****** -Dgbase -h192.168.1.1,192.168.1.2 -vvv <1.sql
--------------
select count(*) from user
--------------


+----------+
| count(*) |
+----------+
|        2 |
+----------+
1 row in set (Elapsed: 00:00:00.01)


--------------
select user from user
--------------


+------------------+
| user             |
+------------------+
| gbase            |
| root             |
```

```
+------------------+
2 rows in set (Elapsed: 00:00:00.00)


Bye
```

4.5 Cluster node management

summary

GBase 8a MPP Cluster supports functions such as cluster expansion, cluster scaling, and cluster node replacement, meeting scenarios such as insufficient data storage space due to increased data storage during cluster operation, single node hardware failure due to prolonged operation, and overall cluster hardware upgrade.

V9.5.3 currently does not support the expansion and contraction of gcware nodes, and supports the replacement of gcware nodes.

4.5.1 Cluster expansion

4.5.1.1 Cluster expansion operation process

- Expanding data nodes within VC:

1. Install cluster software for data nodes that require expansion. Including steps: Modify the demo. options file and →execute the gcinstall script to install the software.
2. Add the expansion node to the specified VC. Including steps: create a new addnode.xml and →execute gcadmin addnodes to add nodes to VC
3. Generate a new distribution for all nodes in the expanded VC, and generate a new datanodemap based on the new distribution. Including steps: Modify gcChangeInfo.xml to →execute gcadmin distribution on all nodes after expansion to generate a new distribution (database username and password must be passed in), →execute initnodeDatamap to generate a new nodeDatamap.
4. Migrate the original cluster data migration to the new nodeDatamap, and delete the old distribution and nodeDatamap of the original cluster. Including steps: Rebalance migration data →gcadmin rmdistribution.

- Expand the gcluster node of the cluster:

Stop the service →modification demo.options of all nodes in the entire cluster and use the gcinstall script to install the cluster software on that node.

- Expand composite nodes (gcluster and gnode are on the same server):

1. Install cluster software for gnode nodes and gcluster nodes that require expansion. Including steps: stop the services of all nodes in the entire cluster, →modify the demo. options file, →execute the gcinstall script to install the software.
2. Add the expansion node to the specified VC. Including steps: create a new addnode.xml and →execute gcadmin addnodes to add nodes to VC
3. Generate a new distribution for all nodes in the expanded VC, and generate a new

datanodemap based on the new distribution. Including steps: Modify gcChangeInfo.xml to →execute gcadmin distribution on all nodes after expansion to generate a new distribution (database username and password must be passed in), →execute initnodeDatamap to generate a new nodeDatamap.

4. Migrate the original cluster data migration to the new nodeDatamap, and delete the old distribution and nodeDatamap of the original cluster. Including steps: Rebalance migration data →gcadmin rmdistribution.

be careful

- V9.5.3 currently only supports the expansion of coordinator nodes and data nodes, and does not support the expansion of gcware.
- The expansion installation operation must be performed using DBA (gbase) users on existing coordinator nodes.
- The expansion installation operation requires applying for a new license file. The license file can be obtained by referring to the section 3.2.1 Obtaining Licenses.
- If the expanded node includes a coordinator node, the cluster service needs to be stopped during installation.
- When expanding the coordinator node installation, if there is a large amount of metadata, it is necessary to increase the timeout time to avoid copying metadata. When executing the gcinstall.py script, add the parameter --timeout=TIMEOUT. The timeout time unit is minutes. If the timeout time is not specified, the default timeout time is 15 minutes.
- If the cluster has a full-text index, the full text will be automatically synchronized to the new node during expansion, and there is no need to install or configure it again. The index configuration and data will be automatically copied, but the index construction requires manual reconstruction.
- Support creating hashmaps based on weights (using hint).
- After configuring a cluster with kerberos authentication, after the expansion is completed, the keytab file and kerberos configuration file should be manually modified by the user according to the actual situation to adapt to the cluster configuration file.
- Full text indexing is not available after expansion and must be rebuilt without the need to copy files.

4.5.1.2 Multi VC mode

4.5.1.2.1 Expanding pure data nodes

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13

Data node:

vc1：172.168.83.11，172.168.83.12

Vc2：172.168.83.13，172.168.83.14

IP of data node to be expanded to vc1: 172.168.83.15

4.5.1.2.1.1Installation node (optional)

If a freenode node already exists, this step can be ignored.

Operating Steps

Step 1: Modify the demo. options file:

1) Set the dataHost parameter to the IP of the node to be installed;
2) Modify the existCoordinateHost parameter to the IP of an existing coordinator node;
3) Modify the existDataHost parameter to the IP addresses of all existing data nodes.

The modified demo. options refer to the following:

```
$ cat demo.options
installPrefix= /opt
#coordinateHost =
#coordinateHostNodeID =1,2,3
dataHost = 172.168.83.15
existCoordinateHost =172.168.83.11,172.168.83.12,172.168.83.13
existDataHost =172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.14
existGcwareHost=172.168.83.11,172.168.83.12,172.168.83.13
#gcwareHost =
#gcwareHostNodeID =
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbasedba'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
```

Step 2: Perform Installation

```
$ ./ gcinstall.py --silent=demo.options
*****************************************************************
***************
Thank you for choosing GBase product!


……

*****************************************************************
Do you accept the above licence agreement ([Y,y]/[N,n])? y
```

```
**********************************************************************
                    Welcome to install GBase products
**********************************************************************
Environmental Checking on gcluster nodes.
CoordinateHost:
DataHost:
172.168.83.15
Are you sure to install GCluster on these nodes ([Y,y]/[N,n])? y

……

172.168.83.15                   install cluster on host 172.168.83.15 successfully.
update and sync configuration file...
Starting all gcluster nodes...
adding new datanodes to gcware...
$
```

##The above message indicates successful installation

The status information after installation is as follows:

```
$ gcadmin
CLUSTER STATE:              ACTIVE

=============================================================
=====
|                        GBASE  COORDINATOR  CLUSTER  INFORMATION
|
=============================================================
=====
|     NodeName     |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
=================================================
|          GBASE VIRTUAL CLUSTER INFORMATION          |
=================================================
|       VcName       | DistributionId |      comment      |
-------------------------------------------------
|         vc1          |            1             | vc1comments |
-------------------------------------------------
|         vc2          |            2             | vc2comments |
```

```
-----------------------------------------------
============================================================
===
|                    GBASE CLUSTER FREE DATA NODE INFORMATION
|
============================================================
===
| NodeName    |     IpAddress      | gnode | syncserver | DataState |
------------------------------------------------------------
| FreeNode1 | 172.168.83.15 | OPEN    |       OPEN       |         0         |
------------------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
1 free data node
```

4.5.1.2.1.2Add nodes to the VC to be expanded

You need to use the addnodes command to add the freenode node to the VC you want to expand before proceeding to the next step.

Operating Steps

Step 1: Modify the gcChangeInfo.xml file:

```
$ cat gcChangeInfo.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
  <rack>
    <node ip="172.168.83.15"/>
  </rack>
</servers>
```

Step 2: Add freenode to vc1:

```
$ gcadmin addnodes gcChangeInfo.xml vc1
gcadmin add nodes ...

flush statemachine success

gcadmin addnodes to vc [vc1] success
After adding, the cluster status information is as follows:
$ gcadmin
CLUSTER STATE:                ACTIVE

============================================================
|                GBASE COORDINATOR CLUSTER INFORMATION                |
============================================================
```

```
|     NodeName     |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
=============================================
|          GBASE VIRTUAL CLUSTER INFORMATION          |
=============================================
|       VcName       | DistributionId |     comment     |
---------------------------------------------
|         vc1          |             1             | vc1comments |
---------------------------------------------
|         vc2          |             2             | vc2comments |
---------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node

$ gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

=============================================
|          GBASE VIRTUAL CLUSTER INFORMATION          |
=============================================
|       VcName       | DistributionId |     comment     |
---------------------------------------------
|         vc1          |             1             | vc1comments |
---------------------------------------------
=============================================================================
======
|                                        VIRTUAL   CLUSTER   DATA   NODE   INFORMATION
|
=============================================================================
======
| NodeName |    IpAddress          | DistributionId | gnode | syncserver | DataState |
-----------------------------------------------------------------------------
|   node1    |172.168.83.11      |            1             | OPEN   |      OPEN      |        0        |
-----------------------------------------------------------------------------
|   node2    |172.168.83.12      |            1             | OPEN   |      OPEN      |        0        |
```

```
--------------------------------------------------------------------------------
|   node3    |172.168.83.15     |                    | OPEN   |     OPEN     |       0      |
3 data node
```

4.5.1.2.1.3Create a new distribution

Operating Steps

Step 1: Modify the gcChangeInfo.xml file in the installation directory, add the node IP to be expanded, and write all node IPs after expansion to the gcChangeInfo.xml file.

The modified gcChangeInfo.xml file reference is as follows:

```
$ cat gcChangeInfo.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
 <rack>
   <node ip="172.168.83.15"/>
   <node ip="172.168.83.11"/>
   <node ip="172.168.83.12"/>
 </rack>
</servers>
```

Step 2: Execute the command to create a distribution.

```
$ gcadmin distribution gcChangeInfo.xml p 1 d 1 db_ user user_ name db_ pwd password vc vc1
gcadmin generate distribution ...

copy system table to 172.168.83.15
gcadmin generate distribution successful
The completed cluster information is as follows:
$ gcadmin showdistribution vc vc1

                    Distribution ID: 3 | State: new | Total segment num: 3

     Primary Segment Node IP       Segment ID       Duplicate Segment node IP
================================================================================
|          172.168.83.15          |       1        |       172.168.83.11          |
--------------------------------------------------------------------------------
|          172.168.83.11          |       2        |       172.168.83.12          |
--------------------------------------------------------------------------------
|          172.168.83.12          |       3        |       172.168.83.15          |
================================================================================
```

```
$ gcadmin distribution gcChangeInfo.xml p 1 d 1 db_ user user_ name db_ pwd password vc vc1
gcadmin generate distribution ...

copy system table to 172.168.83.15
gcadmin generate distribution successful
                    Distribution ID: 1 | State: old | Total segment num: 2

     Primary Segment Node IP       Segment ID       Duplicate Segment node IP
============================================================================
|          172.168.83.11          |       1        |      172.168.83.12                |
----------------------------------------------------------------------------
|          172.168.83.12          |       2        |      172.168.83.11                |
============================================================================
```

4.5.1.2.1.4Initialize hashmap and perform data redistribution

In this step, you can set the priority of the Rebalance task. Set the parameter gcluster first_ rebalancing_ concurrent_ Count=0 prevents the Rebalance task from being executed. Then use the Rebalance instance to add all tables under the current cluster to gclusterdb.rebalancing_ Status. Set gcluster after adjusting the priority of the rebalance task for each table_ rebalancing_ concurrent_ Count is the required number of concurrency, starting to perform data redistribution. Please refer to the chapter for detailed steps to adjust the priority of the Rebalance task.

Operating Steps

Step 1: Initialize hashmap:

```
$ gccli -uroot

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights Reserved.

gbase> use vc vc1;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> initnodedatamap;
Query OK, 0 rows affected, 5 warnings (Elapsed: 00:00:01.45)
```

Step 2: Perform data redistribution, without priority adjustment in this example:

```
gbase> show variables like '%rebalanc%';
+-----------------------------------------------------------+-----------+
| Variable_ name                                            | Value     |
+-----------------------------------------------------------+-----------+
| _ t_ gcluster_ rebalance_ mirror_ node                    | 0         |
| gcluster_ load_ rebalance_ seed                           | 5         |
| gcluster_ rebalancing_ concurrent_ count                  | 5         |
| gcluster_ rebalancing_ ignore_ mirror                     | OFF       |
| gcluster_ rebalancing_ immediate_ recover_ internal_ table | OFF       |
| gcluster_ rebalancing_ parallel_ degree                   | 4         |
| gcluster_ rebalancing_ random_ table_ quick_ mode          | 1         |
| gcluster_ rebalancing_ step                               | 100000000 |
| gcluster_ rebalancing_ update_ status_ on_ drop_ table    | ON        |
+-----------------------------------------------------------+-----------+
9 rows in set (Elapsed: 00:00:00.24)

gbase> rebalance instance;
Query OK, 2 rows affected (Elapsed: 00:00:01.45)
View Rebalance Status:
gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+------------+-----------+------------+
| index_ name | status       | percentage |
+------------+-----------+------------+
| demo.t         | COMPLETED |              100 |
| demo.tt       | COMPLETED |              100 |
+------------+-----------+------------+
2 rows in set (Elapsed: 00:00:00.04)

gbase> quit
Bye
```

4.5.1.2.1.5Delete old distribution

Operating Steps

Step 1: Confirm the current distribution ID. In the current example, the new distribution ID is 3 and the old distribution ID is 1:

```
$ gcadmin showdistribution vc vc1

                Distribution ID: 3 | State: new | Total segment num: 3

     Primary Segment Node IP       Segment ID        Duplicate Segment node IP
==========================================================================
|          172.168.83.15          |       1        |       172.168.83.11          |
--------------------------------------------------------------------------
|          172.168.83.11          |       2        |       172.168.83.12          |
--------------------------------------------------------------------------
|          172.168.83.12          |       3        |       172.168.83.15          |
==========================================================================

                Distribution ID: 1 | State: old | Total segment num: 2

     Primary Segment Node IP       Segment ID        Duplicate Segment node IP
==========================================================================
|          172.168.83.11          |       1        |       172.168.83.12          |
--------------------------------------------------------------------------
|          172.168.83.12          |       2        |       172.168.83.11          |
==========================================================================
```

Step 2: Confirm that there are no tables using the old Distribution ID in the current cluster

```
gbase> select index_ name,tbname,data_ distribution_ id,vc_ id from gbase.table_
distribution;
+------------------------------+-----------------+---------------------+---------+
| index_ name                  |tbname           |data_ distribution_ id | vc_ id
|
+------------------------------+-----------------+---------------------+---------+
| gclusterdb.rebalancing_ status|rebalancing_ status|                   3 | vc00001 |
| gclusterdb.dual              |dual             |                     3 |
vc00001 |
| demo.t                       |t                |                     3 |
vc00001 |
| demo.tt                      |tt               |                     3 |
vc00001 |
+------------------------------+-----------------+---------------------+---------+
```

Step 3: Delete the old distribution:

```
$ gcadmin rmdistribution 1 vc vc1
cluster distribution ID [1]
it will be removed now
please ensure this is ok, input [Y,y] or [N,n]: y
select count(*) from gbase.nodedatamap where data_ distribution_ id=1 result is not 0
refreshnodedatamap drop 1 success
gcadmin remove distribution [1] success
$ gcadmin showdistribution vc vc1

                Distribution ID: 3 | State: new | Total segment num: 3

  Primary Segment Node IP      Segment ID      Duplicate Segment node IP
================================================================================
|       172.168.83.15       |       1       |       172.168.83.11       |
--------------------------------------------------------------------------------
|       172.168.83.11       |       2       |       172.168.83.12       |
--------------------------------------------------------------------------------
|       172.168.83.12       |       3       |       172.168.83.15       |
================================================================================
```

4.5.1.2.2 Expanding pure coordinator nodes

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13

Data node:

vc1：172.168.83.11，172.168.83.12

Vc2：172.168.83.13，172.168.83.14

Add a new coordinator node IP: 172.168.83.15

4.5.1.2.2.1Preparing configuration files

Operating Steps

Step 1: Modify the demo. options file:

1) Set the coordinatiteHost parameter to the IP of the node to be installed;
2) Set the coordinateHostNodeID parameter to the ID set for the node to be installed, which corresponds one-to-one with the coordinateHost node setting and is not a duplicate integer value;

3) Modify the existCoordinateHost parameter to the IP of an existing coordinator node;
4) Modify the existDataHost parameter to the IP addresses of all existing data nodes.

The modified demo. options refer to the following:

```
$ cat demo.options
installPrefix= /opt
coordinateHost = 172.168.83.15
coordinateHostNodeID =15
#dataHost = 172.168.83.15
existCoordinateHost =172.168.83.11,172.168.83.12,172.168.83.13
existDataHost =172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.14
existGcwareHost=172.168.83.11,172.168.83.12,172.168.83.13
#gcwareHost =
#gcwareHostNodeID =
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbasedba'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
```

4.5.1.2.2.2Stop cluster services for all nodes

Operating Steps

Step 1: Execute the cluster service stop command on all nodes

```
$ gcluster_ services all stop
Stopping gcrecover :                                       [  OK  ]
Stopping gcluster :                                        [  OK  ]
Stopping gbase :                                           [  OK  ]
Stopping syncserver :                                      [  OK  ]
$ gcware_ services all stop
Stopping GCWareMonit success!
Stopping gcware :                                          [  OK  ]
```

4.5.1.2.2.3Install nodes

be careful

- If there is a lot of metadata, it is necessary to increase the timeout time to avoid the timeout of copying metadata. When executing the gcinstall.py script, add the parameter -- timeout=TIMEOUT. The timeout time unit is minutes. If the timeout time is not specified, the default timeout time is 15 minutes.

Operating Steps

Step 1: Perform Installation

```
$ ./ gcinstall.py --silent=demo.options --timeout=120
********************************************************************************
Thank you for choosing GBase product!

……

********************************************************************
Do you accept the above licence agreement ([Y,y]/[N,n])? y
********************************************************************
                        Welcome to install GBase products
********************************************************************
Environmental Checking on gcluster nodes.
CoordinateHost:
DataHost:
172.168.83.15
Are you sure to install GCluster on these nodes ([Y,y]/[N,n])? y

……


172.168.83.11           install cluster on host 172.168.83.11 successfully.
172.168.83.12           install cluster on host 172.168.83.12 successfully.
172.168.83.13           install cluster on host 172.168.83.13 successfully.
172.168.83.15           install cluster on host 172.168.83.15 successfully.
update and sync configuration file...
Starting all gcluster nodes...
sync coordinator system tables...
$

##The above message indicates successful installation
The status information after installation is as follows:
$ gcadmin
CLUSTER STATE:          ACTIVE


=============================================================
|              GBASE COORDINATOR CLUSTER INFORMATION              |
=============================================================
|    NodeName    |    IpAddress    | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |   OPEN   |   OPEN   |      0      |
-------------------------------------------------------------
```

```
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator4 | 172.168.83.15 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
=============================================
|        GBASE VIRTUAL CLUSTER INFORMATION        |
=============================================
|       VcName       | DistributionId |    comment    |
---------------------------------------------
| vc1                |           1            |               |
---------------------------------------------
| vc2                |           2            |               |
---------------------------------------------

2 virtual cluster: vc1, vc2
4 coordinator node
0 free data node
```

4.5.1.2.3 Expanding composite nodes

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13

Data node:

vc1：172.168.83.11，172.168.83.12

vc2：172.168.83.13，172.168.83.14

Coordinator nodes to be expanded: 172.168.83.15

IP of data node to be expanded to vc1: 172.168.83.15172.168.83.16

4.5.1.2.3.1Preparing to configure node files

Operating Steps

Step 1: Modify the demo. options file:

1) Set coordinatiteHost as the IP of the management node to be installed;
2) Set the coordinates HostNodeID to the ID set for the management node to be installed, which corresponds one-to-one to the coordinates Host node and is not a duplicate integer value;
3) Set the dataHost parameter to the IP of the node to be installed;

4) Modify the existCoordinateHost parameter to the IP of an existing coordinator node;
5) Modify the existDataHost parameter to the IP addresses of all existing data nodes.

The modified demo. options refer to the following:

```
$ cat demo.options
installPrefix= /opt
coordinateHost =172.168.83.15
coordinateHostNodeID =15
dataHost =172.168.83.15,172.168.83.16
existCoordinateHost =172.168.83.11,172.168.83.12,172.168.83.13
existDataHost =172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.14
existGcwareHost=172.168.83.11,172.168.83.12,172.168.83.13
#gcwareHost =
#gcwareHostNodeID =
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbasedba'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
```

4.5.1.2.3.2Stop cluster services for all nodes

Operating Steps

Step 1: Execute the cluster service stop command on all nodes

```
$ gcluster_ services all stop
Stopping gcrecover :                                       [  OK  ]
Stopping gcluster :                                        [  OK  ]
Stopping gbase :                                           [  OK  ]
Stopping syncserver :                                      [  OK  ]
$ gcware_ services all stop
Stopping GCWareMonit success!
Stopping gcware :                                          [  OK  ]
```

4.5.1.2.3.3Install expansion nodes

Operating Steps

Step 1: Perform Installation

```
$ ./ gcinstall.py --silent=demo.options
************************************************************
***************
Thank you for choosing GBase product!
```

```
.................

*********************************************************************
Do you accept the above licence agreement ([Y,y]/[N,n])? y
*********************************************************************
                        Welcome to install GBase products
*********************************************************************
Environmental Checking on gcluster nodes.
CoordinateHost:
172.168.83.15
DataHost:
172.168.83.15       172.168.83.16
Are you sure to install GCluster on these nodes ([Y,y]/[N,n])? y

.................


172.168.83.11                   install cluster on host 172.168.83.11 successfully.
172.168.83.12                   install cluster on host 172.168.83.12 successfully.
172.168.83.13                   install cluster on host 172.168.83.13 successfully.
172.168.83.15                   install cluster on host 172.168.83.15 successfully.
172.168.83.16                   install cluster on host 172.168.83.16 successfully.
update and sync configuration file...
Starting all gcluster nodes...
sync coordinator system tables...
adding new datanodes to gcware...
$

##The above message indicates successful installation
The status information after installation is as follows:
$ gcadmin
CLUSTER STATE:              ACTIVE

=============================================================
=====
|                         GBASE  COORDINATOR  CLUSTER  INFORMATION
|
=============================================================
=====
|     NodeName     |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
```

```
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator4 | 172.168.83.15 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
=============================================
|        GBASE VIRTUAL CLUSTER INFORMATION         |
=============================================
|       VcName       | DistributionId |    comment    |
---------------------------------------------
| vc1                |            1             |                |
---------------------------------------------
| vc2                |            2             |                |
---------------------------------------------
=============================================================
===
|                        GBASE  CLUSTER  FREE  DATA  NODE  INFORMATION
|
=============================================================
===
| NodeName    |     IpAddress      | gnode | syncserver | DataState |
-------------------------------------------------------------
| FreeNode1 | 172.168.83.15 | OPEN    |       OPEN       |        0        |
-------------------------------------------------------------
| FreeNode2 | 172.168.83.16 | OPEN    |       OPEN       |        0        |
-------------------------------------------------------------

2 virtual cluster: vc1, vc2
4 coordinator node
2 free data node
```

4.5.1.2.3.4Add nodes to the VC to be expanded

Operating Steps

Step 1: Modify the gcAddInfo.xml file and write the two newly added IPs to vc1 into the file:

```
$ cat gcAddInfo.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
 <rack>
  <node ip="172.168.83.15"/>
  <node ip="172.168.83.16"/>
 </rack>
</servers>
```

Step 2: Add freenode to vc1:

```
$ gcadmin addnodes gcAddInfo.xml vc1
gcadmin add nodes ...

flush statemachine success

gcadmin addnodes to vc [vc1] success

After adding, the cluster status information is as follows:
$ gcadmin
CLUSTER STATE:              ACTIVE

=============================================================
|                GBASE COORDINATOR CLUSTER INFORMATION                |
=============================================================
|    NodeName    |    IpAddress    | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator4 | 172.168.83.15 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
=============================================
|       GBASE VIRTUAL CLUSTER INFORMATION       |
=============================================
|      VcName      | DistributionId |   comment   |
---------------------------------------------
|       vc1        |          1          |               |
---------------------------------------------
|       vc2        |          2          |               |
---------------------------------------------

2 virtual cluster: vc1, vc2
4 coordinator node
0 free data node

$ gcadmin showcluster vc vc1
CLUSTER STATE:              ACTIVE
VIRTUAL CLUSTER MODE:   NORMAL
```

```
==============================================
|        GBASE VIRTUAL CLUSTER INFORMATION        |
==============================================
|      VcName       | DistributionId |    comment    |
--------------------------------------------
|        vc1          |             1             |                    |
--------------------------------------------
============================================================================
===
|                          VIRTUAL CLUSTER DATA NODE INFORMATION                                  |
============================================================================
===
| NodeName |    IpAddress      | DistributionId | gnode | syncserver | DataState |
---------------------------------------------------------------------------
|   node1     |172.168.83.11 |            1              | OPEN   |       OPEN       |         0         |
---------------------------------------------------------------------------
|   node2     |172.168.83.12 |            1              | OPEN   |       OPEN       |         0         |
---------------------------------------------------------------------------
|   node3     |172.168.83.15 |                           | OPEN   |       OPEN       |         0         |
---------------------------------------------------------------------------
|   node4     |172.168.83.16 |                           | OPEN   |       OPEN       |         0         |
---------------------------------------------------------------------------
4 data node
```

4.5.1.2.3.5Create a new distribution

Operating Steps

Step 1: Modify the gcChangeInfo.xml file in the installation directory, add the node IP to be expanded, and write all node IPs after expansion to the gcChangeInfo.xml file.

The modified gcChangeInfo.xml file reference is as follows:

```
$ cat gcChangeInfo.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
  <rack>
    <node ip="172.168.83.11"/>
    <node ip="172.168.83.12"/>
    <node ip="172.168.83.15"/>
    <node ip="172.168.83.16"/>
  </rack>
</servers>
```

Step 2: Execute the command to create a distribution.

```
$ gcadmin distribution gcChangeInfo.xml p 1 d 1 db_ user user_ name db_ pwd password vc vc1
gcadmin generate distribution ...

NOTE: node [172.168.83.15] is coordinator node, it shall be data node too
copy system table to 172.168.83.16
copy system table to 172.168.83.15
gcadmin generate distribution successful


The completed cluster information is as follows:
$ gcadmin showdistribution vc vc1

      Distribution ID: 3 | State: new | Total segment num: 4

    Primary Segment Node IP     Segment ID     Duplicate Segment node IP
=============================================================================
|          172.168.83.11          |        1         |          172.168.83.12          |
-----------------------------------------------------------------------------
|          172.168.83.12          |        2         |          172.168.83.15          |
-----------------------------------------------------------------------------
|          172.168.83.15          |        3         |          172.168.83.16          |
-----------------------------------------------------------------------------
|          172.168.83.16          |        4         |          172.168.83.11          |
=============================================================================

      Distribution ID: 1 | State: old | Total segment num: 2

    Primary Segment Node IP      Segment ID     Duplicate Segment node IP
=============================================================================
|          172.168.83.11          |        1         |          172.168.83.12          |
-----------------------------------------------------------------------------
|          172.168.83.12          |        2         |          172.168.83.11          |
=============================================================================
```

4.5.1.2.3.6Initialize hashmap and perform data redistribution

In this step, you can set the priority of the Rebalance task. Set the parameter gcluster first_ rebalancing_ concurrent_ Count=0 prevents the Rebalance task from being executed. Then use the Rebalance instance to add all tables under the current cluster to gclusterdb.rebalancing_ Status. Set gcluster after adjusting the priority of the rebalance task for each table_ rebalancing_ concurrent_ Count is the required number of concurrency, starting to perform data redistribution. Please refer to the chapter for detailed steps to adjust the priority of the Rebalance task.

Operating Steps

Step 1: Initialize hashmap:

```
$ gccli -uroot

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights Reserved.

gbase> use vc vc1;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> initnodedatamap;
Query OK, 0 rows affected, 5 warnings (Elapsed: 00:00:01.45)
```

Step 2: Perform data redistribution:

```
gbase> show variables like '%rebalanc%';
+-----------------------------------------------------------+-----------+
| Variable_ name                                            | Value     |
+-----------------------------------------------------------+-----------+
| _ t_ gcluster_ rebalance_ mirror_ node                    | 0         |
| gcluster_ load_ rebalance_ seed                           | 5         |
| gcluster_ rebalancing_ concurrent_ count                  | 5         |
| gcluster_ rebalancing_ ignore_ mirror                     | OFF       |
| gcluster_ rebalancing_ immediate_ recover_ internal_ table | OFF       |
| gcluster_ rebalancing_ parallel_ degree                   | 4         |
| gcluster_ rebalancing_ random_ table_ quick_ mode         | 1         |
| gcluster_ rebalancing_ step                               | 100000000 |
| gcluster_ rebalancing_ update_ status_ on_ drop_ table    | ON        |
+-----------------------------------------------------------+-----------+
9 rows in set (Elapsed: 00:00:00.24)

gbase> rebalance database demo;
Query OK, 2 rows affected (Elapsed: 00:00:01.45)
View Rebalance Status:
gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_ status;
+------------+-----------+------------+
| index_ name | status      | percentage |
+------------+-----------+------------+
| demo.t       | COMPLETED |          100 |
| demo.tt      | COMPLETED |          100 |
+------------+-----------+------------+
2 rows in set (Elapsed: 00:00:00.04)

gbase> quit
Bye
```

4.5.1.2.3.7Delete old distribution

Operating Steps

Step 1: Confirm the current distribution ID. In the current example, the new distribution ID is 3 and the old distribution ID is 1:

```
$ gcadmin showdistribution vc vc1

                    Distribution ID: 3 | State: new | Total segment num: 4

  Primary Segment Node IP          Segment ID          Duplicate Segment node IP
==========================================================================================
|       172.168.83.11       |          1          |       172.168.83.12       |
------------------------------------------------------------------------------
|       172.168.83.12       |          2          |       172.168.83.15       |
------------------------------------------------------------------------------
|       172.168.83.15       |          3          |       172.168.83.16       |
------------------------------------------------------------------------------
|       172.168.83.16       |          4          |       172.168.83.11       |
==========================================================================================

                    Distribution ID: 1 | State: old | Total segment num: 2

  Primary Segment Node IP          Segment ID          Duplicate Segment node IP
==========================================================================================
|       172.168.83.11       |          1          |       172.168.83.12       |
------------------------------------------------------------------------------
|       172.168.83.12       |          2          |       172.168.83.11       |
==========================================================================================
```

Step 2: Confirm that there are no tables using the old Distribution ID in the current cluster

```
gbase> select index_ name,tbname,data_ distribution_ id,vc_ id from gbase.table_ distribution;
+------------------------------+-----------------+---------------------+---------+
| index_ name                  |tbname           |data_ distribution_ id | vc_ id      |
+------------------------------+-----------------+---------------------+---------+
| gclusterdb.rebalancing_ status|rebalancing_ status|                   3 | vc00001 |
| gclusterdb.dual              |dual             |                     3 | vc00001 |
| demo.t                       |t                |                     3 | vc00001 |
| demo.tt                      |tt               |                     3 | vc00001 |
+------------------------------+-----------------+---------------------+---------+
```

Step 3: Delete the old distribution:

```
$ gcadmin rmdistribution 1 vc vc1
cluster distribution ID [1]
it will be removed now
please ensure this is ok, input [Y,y] or [N,n]: y
select count(*) from gbase.nodedatamap where data_ distribution_ id=1 result is not 0
refreshnodedatamap drop 1 success
gcadmin remove distribution [1] success
$ gcadmin showdistribution vc vc1
$ gcadmin showdistribution vc vc1

        Distribution ID: 3 | State: new | Total segment num: 4

   Primary Segment Node IP          Segment ID          Duplicate Segment node IP
================================================================================
|        172.168.83.11        |          1          |        172.168.83.12        |
--------------------------------------------------------------------------------
|        172.168.83.12        |          2          |        172.168.83.15        |
--------------------------------------------------------------------------------
|        172.168.83.15        |          3          |        172.168.83.16        |
--------------------------------------------------------------------------------
|        172.168.83.16        |          4          |        172.168.83.11        |
================================================================================
```

4.5.1.3 Compatibility mode

4.5.1.3.1 Expanding pure data nodes

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13172.168.83.14

Data node:

vc1：172.168.83.11，172.168.83.12，172.168.83.13

IP of data node to be expanded to vc1: 172.168.83.15

4.5.1.3.1.1Install nodes

Operating Steps

Step 1: Modify the demo. options file:

1) Set the dataHost parameter to the IP of the node to be installed;
2) Modify the existCoordinateHost parameter to the IP of an existing coordinator node;
3) Modify the existDataHost parameter to the IP addresses of all existing data nodes.

The modified demo. options refer to the following:

```
$ cat demo.options
installPrefix= /opt
#coordinateHost =172.168.83.14
#coordinateHostNodeID = 14
dataHost =172.168.83.15
existCoordinateHost =172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.14
existDataHost =172.168.83.11,172.168.83.12,172.168.83.13
existGcwareHost=172.168.83.11,172.168.83.12,172.168.83.13
#gcwareHost =
#gcwareHostNodeID =
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbasedba'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
```

Step 2: Perform Installation

```
$ ./ gcinstall.py --silent=demo.options
************************************************************************
***************
Thank you for choosing GBase product!

……

************************************************************************
Do you accept the above licence agreement ([Y,y]/[N,n])? y
************************************************************************
                        Welcome to install GBase products
************************************************************************
Environmental Checking on gcluster nodes.
CoordinateHost:
DataHost:
172.168.83.15
Are you sure to install GCluster on these nodes ([Y,y]/[N,n])? y

……


172.168.83.15                install cluster on host 172.168.83.15 successfully.
```

```
update and sync configuration file...
Starting all gcluster nodes...
adding new datanodes to gcware...
$

##The above message indicates successful installation
The status information after installation is as follows:
$ gcadmin
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL


=============================================================
======
|                          GBASE  COORDINATOR  CLUSTER  INFORMATION
|
=============================================================
======
|     NodeName     |      IpAddress      | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator4 | 172.168.83.14 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
=============================================================
=======
|                                  GBASE  DATA  CLUSTER  INFORMATION
|
=============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-------------------------------------------------------------
| node1    |172.168.83.11|          1          |OPEN |     OPEN     |      0      |
-------------------------------------------------------------
| node2    |172.168.83.12|          1          |OPEN |     OPEN     |      0      |
-------------------------------------------------------------
| node3    |172.168.83.13|          1          |OPEN |     OPEN     |      0      |
-------------------------------------------------------------
| node4    |172.168.83.15|                     |OPEN |     OPEN     |      0      |
-------------------------------------------------------------
```

4.5.1.3.1.2Create a new distribution

Operating Steps

Step 1: Modify the gcChangeInfo.xml file in the installation directory, add the node IP to be expanded, and write all node IPs after expansion to the gcChangeInfo.xml file.

The modified gcChangeInfo.xml file reference is as follows:

```
$ cat gcChangeInfo.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
 <rack>
   <node ip="172.168.83.11"/>
   <node ip="172.168.83.12"/>
   <node ip="172.168.83.13"/>
   <node ip="172.168.83.15"/>
 </rack>
</servers>
```

Step 2: Execute the command to create a distribution.

```
$ gcadmin distribution gcChangeInfo.xml p 1 d 1
gcadmin generate distribution ...

copy system table to 172.168.83.15
gcadmin generate distribution successful
The completed cluster information is as follows:
$ gcadmin showdistribution

      Distribution ID: 2 | State: new | Total segment num: 4

    Primary Segment Node IP     Segment ID     Duplicate Segment node IP
========================================================================
|          172.168.83.11         |       1        |          172.168.83.12          |
------------------------------------------------------------------------
|          172.168.83.12         |       2        |          172.168.83.13          |
------------------------------------------------------------------------
|          172.168.83.13         |       3        |          172.168.83.15          |
------------------------------------------------------------------------
|          172.168.83.15         |       4        |          172.168.83.11          |
========================================================================

      Distribution ID: 1 | State: old | Total segment num: 3
```

```
$ gcadmin distribution gcChangeInfo.xml p 1 d 1
gcadmin generate distribution ...

copy system table to 172.168.83.15
gcadmin generate distribution successful

     Primary Segment Node IP      Segment ID     Duplicate Segment node IP
=============================================================================
|          172.168.83.11          |       1         |          172.168.83.12          |
-----------------------------------------------------------------------------
|          172.168.83.12          |       2         |          172.168.83.13          |
-----------------------------------------------------------------------------
|          172.168.83.13          |       3         |          172.168.83.11          |
=============================================================================
```

4.5.1.3.1.3Initialize hashmap and perform data redistribution

In this step, you can set the priority of the Rebalance task. Set the parameter gcluster first_ rebalancing_ concurrent_ Count=0 prevents the Rebalance task from being executed. Then use the Rebalance instance to add all tables under the current cluster to gclusterdb.rebalancing_ Status. Set gcluster after adjusting the priority of the rebalance task for each table_ rebalancing_ concurrent_ Count is the required number of concurrency, starting to perform data redistribution. Please refer to the chapter for detailed steps to adjust the priority of the Rebalance task.

Operating Steps

Step 1: Initialize hashmap:

```
$ gccli -uroot

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights Reserved.

gbase> initnodedatamap;
Query OK, 0 rows affected, 7 warnings (Elapsed: 00:00:01.45)
```

Step 2: Perform data redistribution:

```
gbase> rebalance instance;
Query OK, 3 rows affected (Elapsed: 00:00:01.45)
View Rebalance Status:
```

```
gbase> rebalance instance;
Query OK, 3 rows affected (Elapsed: 00:00:01.45)
gbase> select index_ name, status, percentage from gclusterdb.rebalancing_ status;
+------------+-----------+------------+
| index_ name | status | percentage |
+------------+-----------+------------+
| demo.t | COMPLETED | 100 |
| demo.ttt | COMPLETED | 100 |
| demo.tt | COMPLETED | 100 |
+------------+-----------+------------+
3 rows in set (Elapsed: 00:00:00.24)

gbase> quit
Bye
```

4.5.1.3.1.4Delete old distribution

Operating Steps

Step 1: Confirm the current distribution ID. In the current example, the new distribution ID is 2 and the old distribution ID is 1:

```
$ gcadmin showdistribution

      Distribution ID: 2 | State: new | Total segment num: 4

    Primary Segment Node IP     Segment ID     Duplicate Segment node IP
=============================================================================
|          172.168.83.11          |       1         |          172.168.83.12          |
-----------------------------------------------------------------------------
|          172.168.83.12          |       2         |          172.168.83.13          |
-----------------------------------------------------------------------------
|          172.168.83.13          |       3         |          172.168.83.15          |
-----------------------------------------------------------------------------
|          172.168.83.15          |       4         |          172.168.83.11          |
=============================================================================

      Distribution ID: 1 | State: old | Total segment num: 3

    Primary Segment Node IP     Segment ID     Duplicate Segment node IP
=============================================================================
|          172.168.83.11          |       1         |          172.168.83.12          |
-----------------------------------------------------------------------------
|          172.168.83.12          |       2         |          172.168.83.13          |
-----------------------------------------------------------------------------
|          172.168.83.13          |       3         |          172.168.83.11          |
=============================================================================
```

Step 2: Confirm that there are no tables using the old Distribution ID in the current cluster

```
gbase> select index_ name,tbname,data_ distribution_ id,vc_ id from
gbase.table_ distribution;
+------------------------------+-----------------+---------------------+---------+
| index_ name                  |tbname           |data_ distribution_ id |
vc_ id     |
+------------------------------+-----------------+---------------------+---------+
| gclusterdb.rebalancing_ status|rebalancing_ status|                   2 |
vc00001 |
| gclusterdb.dual              |dual             |                     2 |
vc00001 |
|  demo.t                                        |t                      |
2 | vc00001 |
| demo.tt                      |tt               |                     2
| vc00001 |
| demo.ttt                     |ttt              |                     2 |
vc00001 |
+------------------------------+-----------------+---------------------+---------+
5 rows in set (Elapsed: 00:00:00.00)
```

Step 3: Delete the old distribution:

```
$ gcadmin rmdistribution 1
cluster distribution ID [1]
it will be removed now
please ensure this is ok, input [Y,y] or [N,n]: y
select count(*) from gbase.nodedatamap where data_ distribution_ id=1 result is not 0
refreshnodedatamap drop 1 success
gcadmin remove distribution [1] success
$ gcadmin showdistribution

      Distribution ID: 2 | State: new | Total segment num: 4

    Primary Segment Node IP     Segment ID     Duplicate Segment node IP
============================================================================
|         172.168.83.11         |       1        |         172.168.83.12         |
----------------------------------------------------------------------------
|         172.168.83.12         |       2        |         172.168.83.13         |
----------------------------------------------------------------------------
|         172.168.83.13         |       3        |         172.168.83.15         |
----------------------------------------------------------------------------
|         172.168.83.15         |       4        |         172.168.83.11         |
============================================================================
```

4.5.1.3.2 Expanding pure coordinator nodes

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13

Data node: 172.168.83.11172.168.83.12172.168.83.13

Add a new coordinator node IP: 172.168.83.14

4.5.1.3.2.1Preparing configuration files

Operating Steps

Step 1: Modify the demo. options file:

1) Set the coordinatiteHost parameter to the IP of the node to be installed;
2) Set the coordinateHostNodeID parameter to the ID set for the node to be installed, which corresponds one-to-one with the coordinateHost node setting and is not a duplicate integer value;

3) Modify the existCoordinateHost parameter to the IP of an existing coordinator node;
4) Modify the existDataHost parameter to the IP addresses of all existing data nodes.

The modified demo. options refer to the following:

```
$ cat demo.options
installPrefix= /opt
coordinateHost =172.168.83.14
coordinateHostNodeID = 14
#dataHost =
existCoordinateHost =172.168.83.11,172.168.83.12,172.168.83.13
existDataHost =172.168.83.11,172.168.83.12,172.168.83.13
existGcwareHost=172.168.83.11,172.168.83.12,172.168.83.13
#gcwareHost =
#gcwareHostNodeID =
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbasedba'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
```

4.5.1.3.2.2Stop cluster services for all nodes

Operating Steps

Step 1: Execute the cluster service stop command on all nodes

```
$ gcluster_ services all stop
Stopping gcrecover :                                       [  OK  ]
Stopping gcluster :                                        [  OK  ]
Stopping gbase :                                           [  OK  ]
Stopping syncserver :                                      [  OK  ]
$ gcware_ services all stop
Stopping GCWareMonit success!
Stopping gcware :                                          [  OK  ]
```

4.5.1.3.2.3Install nodes

be careful

- If there is a lot of metadata, it is necessary to increase the timeout time to avoid the timeout of copying metadata. When executing the gcinstall.py script, add the parameter -- timeout=TIMEOUT. The timeout time unit is minutes. If the timeout time is not specified, the default timeout time is 15 minutes.

Operating Steps

Step 1: Perform Installation

```
$ ./ gcinstall.py --silent=demo.options --timeout=120
*****************************************************************************
Thank you for choosing GBase product!

……

*********************************************************************
Do you accept the above licence agreement ([Y,y]/[N,n])? y
*********************************************************************
                        Welcome to install GBase products
*********************************************************************
Environmental Checking on gcluster nodes.
CoordinateHost:
172.168.83.14
DataHost:
Are you sure to install GCluster on these nodes ([Y,y]/[N,n])? y

……


172.168.83.11           install cluster on host 172.168.83.11 successfully.
172.168.83.12           install cluster on host 172.168.83.12 successfully.
172.168.83.13           install cluster on host 172.168.83.13 successfully.
172.168    .83.14       install cluster on host 172.168.83.14 successfully.
update and sync configuration file...
Starting all gcluster nodes...
sync coordinator system tables...
$
```

##The above message indicates successful installation

The status information after installation is as follows:

```
$ gcadmin
CLUSTER STATE:          ACTIVE
VIRTUAL CLUSTER MODE:   NORMAL

=============================================================
|              GBASE COORDINATOR CLUSTER INFORMATION        |
=============================================================
|   NodeName   |   IpAddress   | gcware | gcluster | DataState |
-------------------------------------------------------------
```

```
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator4 | 172.168.83.14 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
=============================================================================
==
|                          GBASE DATA CLUSTER INFORMATION                                    |
=============================================================================
==
| NodeName |    IpAddress    | DistributionId | gnode | syncserver | DataState |
---------------------------------------------------------------------------
|   node1    |172.168.83.11|          1          | OPEN   |     OPEN     |       0       |
---------------------------------------------------------------------------
|   node2    |172.168.83.12|          1          | OPEN   |     OPEN     |       0       |
---------------------------------------------------------------------------
|   node3    |172.168.83.13|          1          | OPEN   |     OPEN     |       0       |
---------------------------------------------------------------------------
```

4.5.1.3.3 Expanding composite nodes

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13172.168.83.14

Data node: 172.168.83.11172.168.83.12172.168.83.13172.168.83.15

Composite nodes to be expanded: 172.168.83.16

4.5.1.3.3.1Preparing to configure node files

Operating Steps

Step 1: Modify the demo. options file:

1) Set coordinatiteHost as the IP of the management node to be installed;
2) Set the coordinates HostNodeID to the ID set for the management node to be installed, which corresponds one-to-one to the coordinates Host node and is not a duplicate integer value;
3) Set the dataHost parameter to the IP of the node to be installed;
4) Modify the existCoordinateHost parameter to the IP of an existing coordinator node;
5) Modify the existDataHost parameter to the IP addresses of all existing data nodes.

The modified demo. options refer to the following:

```
$ cat demo.options
installPrefix= /opt
coordinateHost =172.168.83.16
coordinateHostNodeID = 16
dataHost =172.168.83.16
existCoordinateHost =172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.14
existDataHost =172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.15
existGcwareHost=172.168.83.11,172.168.83.12,172.168.83.13
#gcwareHost =
#gcwareHostNodeID =
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbasedba'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
```

4.5.1.3.3.2Stop cluster services for all nodes

Operating Steps

Step 1: Execute the cluster service stop command on all nodes

```
$ gcluster_ services all stop
Stopping gcrecover :                                       [  OK  ]
Stopping gcluster :                                        [  OK  ]
Stopping gbase :                                           [  OK  ]
Stopping syncserver :                                      [  OK  ]
$ gcware_ services all stop
Stopping GCWareMonit success!
Stopping gcware :                                          [  OK  ]
```

4.5.1.3.3.3Install expansion nodes

Operating Steps

Step 1: Perform Installation

```
$ ./ gcinstall.py --silent=demo.options
***************************************************************
***************
Thank you for choosing GBase product!


……


***************************************************************
Do you accept the above licence agreement ([Y,y]/[N,n])? y
***************************************************************
```

```
                    Welcome to install GBase products
*****************************************************************
Environmental Checking on gcluster nodes.
CoordinateHost:
172.168.83.16
DataHost:
172.168.83.16
Are you sure to install GCluster on these nodes ([Y,y]/[N,n])? y
        172.168.83.11           start install on host 172.168.83.11
172.168.83.12                   start install on host 172.168.83.12
        172.168.83.13           start install on host 172.168.83.13
        172.168.83.14           start install on host 172.168.83.14
        172.168.83.16           start install on host 172.168.83.16
.................


172.168.83.11                   install cluster on host 172.168.83.11 successfully.
172.168.83.12                   install cluster on host 172.168.83.12 successfully.
172.168.83.13                   install cluster on host 172.168.83.13 successfully.
172.168.83.14                   install cluster on host 172.168.83.14 successfully.
172.168.83.16                   install cluster on host 172.168.83.16 successfully.
update and sync configuration file...
Starting all gcluster nodes...
sync coordinator system tables...
adding new datanodes to gcware...
$
```

##The above message indicates successful installation

The status information after installation is as follows:

```
$ gcadmin
CLUSTER STATE:              ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

=============================================================
======
|                         GBASE  COORDINATOR  CLUSTER  INFORMATION
|
=============================================================
======
|    NodeName     |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
```

```
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator4 | 172.168.83.14 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator5 | 172.168.83.16 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
============================================================
=======
|                                GBASE  DATA  CLUSTER  INFORMATION
|
============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-------------------------------------------------------------------
| node1    |172.168.83.11|          2          |OPEN |     OPEN     |      0      |
-------------------------------------------------------------------
| node2    |172.168.83.12|          2          |OPEN |     OPEN     |      0      |
-------------------------------------------------------------------
| node3    |172.168.83.13|          2          |OPEN |     OPEN     |      0      |
-------------------------------------------------------------------
| node4    |172.168.83.15|          2          |OPEN |     OPEN     |      0      |
-------------------------------------------------------------------
| node5    |172.168.83.16|                     |OPEN |     OPEN     |      0      |
-------------------------------------------------------------------
```

4.5.1.3.3.4Create a new distribution

Operating Steps

Step 1: Modify the gcChangeInfo.xml file in the installation directory, add the node IP to be expanded, and write all node IPs after expansion to the gcChangeInfo.xml file.

The modified gcChangeInfo.xml file reference is as follows:

```
$ cat gcChangeInfo.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
 <rack>
   <node ip="172.168.83.11"/>
   <node ip="172.168.83.12"/>
   <node ip="172.168.83.13"/>
   <node ip="172.168.83.15"/>
   <node ip="172.168.83.16"/>
 </rack>
</servers>
```

Step 2: Execute the command to create a distribution.

```
$ gcadmin distribution gcChangeInfo.xml p 1 d 1
gcadmin generate distribution ...

NOTE: node [172.168.83.16] is coordinator node, it shall be data node too
copy system table to 172.168.83.16
gcadmin generate distribution successful
```

The completed cluster information is as follows:

```
$ gcadmin distribution gcChangeInfo.xml p 1 d 1
gcadmin generate distribution ...

NOTE: node [172.168.83.16] is coordinator node, it shall be data node too
copy system table to 172.168.83.16
gcadmin generate distribution successful

$ gcadmin showdistribution


      Distribution ID: 3 | State: new | Total segment num: 5


    Primary Segment Node IP     Segment ID     Duplicate Segment node IP
==========================================================================
|          172.168.83.11          |       1        |          172.168.83.12          |
--------------------------------------------------------------------------
|          172.168.83.12          |       2        |          172.168.83.13          |
--------------------------------------------------------------------------
|          172.168.83.13          |       3        |          172.168.83.15          |
--------------------------------------------------------------------------
|          172.168.83.15          |       4        |          172.168.83.16          |
--------------------------------------------------------------------------
|          172.168.83.16          |       5        |          172.168.83.11          |
==========================================================================


      Distribution ID: 2 | State: old | Total segment num: 4


    Primary Segment Node IP     Segment ID     Duplicate Segment node IP
==========================================================================
|          172.168.83.11          |       1        |          172.168.83.12          |
--------------------------------------------------------------------------
|          172.168.83.12          |       2        |          172.168.83.13          |
--------------------------------------------------------------------------
|          172.168.83.13          |       3        |          172.168.83.15          |
--------------------------------------------------------------------------
|          172.168.83.15          |       4        |          172.168.83.11          |
==========================================================================
```

4.5.1.3.3.5Initialize hashmap and perform data redistribution

In this step, you can set the priority of the Rebalance task. Set the parameter gcluster first_ rebalancing_ concurrent_ Count=0 prevents the Rebalance task from being executed. Then use the Rebalance instance to add all tables under the current cluster to gclusterdb.rebalancing_ Status. Set gcluster after adjusting the priority of the rebalance task for each table_ rebalancing_ concurrent_ Count is the required number of

concurrency, starting to perform data redistribution. Please refer to the chapter for detailed steps to adjust the priority of the Rebalance task.

Operating Steps

Step 1: Initialize hashmap:

```
$ gccli -uroot

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.    All Rights Reserved.
gbase> initnodedatamap;
Query OK, 0 rows affected, 9 warnings (Elapsed: 00:00:01.45)
```

Step 2: Perform data redistribution:

```
gbase> rebalance instance;
Query OK, 3 rows affected (Elapsed: 00:00:01.45)
View Rebalance Status:
gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_ status;
+------------+-----------+------------+
| index_ name | status       | percentage |
+------------+-----------+------------+
| demo.ttt     | COMPLETED |              100 |
| demo.t         | COMPLETED |               100 |
| demo.tt       | COMPLETED |              100 |
+------------+-----------+------------+
3 rows in set (Elapsed: 00:00:00.07)

gbase> quit
Bye
```

4.5.1.3.3.6Delete old distribution

Operating Steps

Step 1: Confirm the current distribution ID. In the current example, the new distribution ID is 3 and the old distribution ID is 2:

```
$ gcadmin showdistribution

      Distribution ID: 3 | State: new | Total segment num: 5

    Primary Segment Node IP     Segment ID     Duplicate Segment node IP
=============================================================================
|          172.168.83.11          |       1         |          172.168.83.12          |
-----------------------------------------------------------------------
|          172.168.83.12          |       2         |          172.168.83.13          |
-----------------------------------------------------------------------
|          172.168.83.13          |       3         |          172.168.83.15          |
-----------------------------------------------------------------------
|          172.168.83.15          |       4         |          172.168.83.16          |
-----------------------------------------------------------------------
|          172.168.83.16          |       5         |          172.168.83.11          |
=============================================================================

      Distribution ID: 2 | State: old | Total segment num: 4

    Primary Segment Node IP     Segment ID     Duplicate Segment node IP
=============================================================================
|          172.168.83.11          |       1         |          172.168.83.12          |
-----------------------------------------------------------------------
|          172.168.83.12          |       2         |          172.168.83.13          |
-----------------------------------------------------------------------
|          172.168.83.13          |       3         |          172.168.83.15          |
-----------------------------------------------------------------------
|          172.168.83.15          |       4         |          172.168.83.11          |
=============================================================================
```

Step 2: Confirm that there are no tables using the old Distribution ID in the current cluster

```
gbase> select index_ name,tbname,data_ distribution_ id,vc_ id from gbase.table_ distribution;
+------------------------------+-----------------+----------------------+---------+
| index_ name                  |tbname           |data_ distribution_ id | vc_ id  |
+------------------------------+-----------------+----------------------+---------+
| gclusterdb.rebalancing_ status|rebalancing_ status|                    3 | vc00001 |
| gclusterdb.dual              |dual             |                      3 | vc00001 |
| demo.t                       |t                |                      3 | vc00001 |
| demo.tt                      |tt               |                      3 | vc00001 |
| demo.ttt                     |ttt              |                      3 | vc00001 |
+------------------------------+-----------------+----------------------+---------+
5 rows in set (Elapsed: 00:00:00.00)
```

Step 3: Delete the old distribution:

```
$ gcadmin rmdistribution 2
cluster distribution ID [2]
it will be removed now
please ensure this is ok, input [Y,y] or [N,n]: y
select count(*) from gbase.nodedatamap where data_ distribution_ id=2 result is not 0
refreshnodedatamap drop 2 success
gcadmin remove distribution [2] success$ gcadmin showdistribution vc vc1
$ gcadmin showdistribution

      Distribution ID: 3 | State: new | Total segment num: 5

  Primary Segment Node IP       Segment ID     Duplicate Segment node IP
=========================================================================
|       172.168.83.11          |       1       |       172.168.83.12          |
-------------------------------------------------------------------------
|       172.168.83.12          |       2       |       172.168.83.13          |
-------------------------------------------------------------------------
|       172.168.83.13          |       3       |       172.168.83.15          |
-------------------------------------------------------------------------
|       172.168.83.15          |       4       |       172.168.83.16          |
-------------------------------------------------------------------------
|       172.168.83.16          |       5       |       172.168.83.11          |
=========================================================================
```

4.5.2 Cluster scaling

be careful

- Shrinkage containing data nodes requires two data redistribution operations
  - The first data redistribution only modifies the distribution strategy and does not result in true data distribution
  - The second data redistribution will result in a true data redistribution

4.5.2.1 Operation process of volume reduction

- Shrink the data nodes in VC:

1. Redistributing data, transferring data from the data nodes to be deleted to other nodes, and clearing data from the data nodes to be deleted. Including steps: Create a new distribution (excluding nodes that will be deleted) Create a new nodeDatamap (initnodeDatamap) →based on the new distribution, →redistribute data to the new nodeDatamap, and →delete the old nodeDatamap and distribution.
2. Remove the node from the cluster. Including steps: Remove the node from VC and become a free node of the cluster, completely delete the node →from the cluster.
3. Uninstall the cluster software on this node. The steps include stopping the gnode service on the deleted node, →modifying demo. options, and uninstalling the software using the uninstall script.

   Note: The gnode node service includes gbase service and syncserver service. If there is still a gcluster node on the gnode node server and the gcluster node needs to be retained, stopping the service does not require direct use of gcluster_ Services all stop To stop all services, simply use the following command to stop each service individually:

   gcmonit.sh stop

   gcluster_ services gbase_ IP stop, such as gcluster_ services gbase_ 192.168.146.40 stop

   gcluster_ services syncserver_ IP stop, such as gcluster_ services syncserver_ 192.168.146.40 stop

- The gcluster node of the downsizing cluster:

Stop the service →modification demo.options of all nodes in the entire cluster and uninstall the cluster software on that node using the uninstall script.

- Shrinkage composite node (gcluster and gnode are on the same server):

1. Redistributing data, transferring data from the data nodes to be deleted to other nodes, and clearing data from the data nodes to be deleted.
2. Remove the node from the cluster. Including steps: Remove the node from VC and become a freenode of the cluster. Completely delete the node →from the cluster
3. Stop the services of all nodes in the entire cluster
4. Modify demo. options to uninstall the cluster software on the server using the uninstall script (including gcluster and gnode nodes)

- Shrink the entire VC step:

Delete all library tables in the VC that need to be scaled down and →delete the VC

Please refer to 4.3.1.3.2 for deleting virtual clusters

 **be careful**

- V9.5.3 does not support gcware node scaling;
- The shrink operation must be performed using a DBA user (gbase) on an existing coordinator node;
- To uninstall a data node, you only need to perform the operation of stopping cluster node services on the uninstalled node;
- Shrinking a cluster can unload nodes, or it is not necessary to unload nodes. Instead, the nodes are kept in the freenode state and can be transferred to other VC for use
- When using the uninstall command to remove nodes, do not use the force parameter to avoid not checking whether the node to be uninstalled is being used by the cluster during uninstallation, resulting in data corruption.

4.5.2.2 Multi VC mode

4.5.2.2.1 Pure data node scaling

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13172.168.83.15

Data node:

vc1：172.168.83.11，172.168.83.12，172.168.83.15，172.168.83.16

Vc2：172.168.83.13，172.168.83.14

IP of data node to be scaled down: 172.168.83.16

4.5.2.2.1.1Create a new distribution

Operating Steps

Step 1: Copy gcChangeInfo.xml from the installation directory to gcS_ VC1. XML file, and then remove the node IP to be shrunk.

Modified gcS_ The vc1. XML file reference is as follows:

```
$cp gcChangeInfo.xml gcS_ vc1.xml
$cat gcS_ vc1.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
 <rack>
  <node ip="172.168.83.11"/>
  <node ip="172.168.83.12"/>
  <node ip="172.168.83.15"/>
 </rack>
</servers>
```

Step 2: Execute the command to create a distribution.

```
$ gcadmin distribution gcS_ vc1.xml p 1 d 1 vc vc1
gcadmin generate distribution ...

gcadmin generate distribution successful
```

The completed cluster information is as follows:

```
$ gcadmin showdistribution vc vc1

                Distribution ID: 4 | State: new | Total segment num: 3

  Primary Segment Node IP        Segment ID        Duplicate Segment node IP
=============================================================================
|      172.168.83.11      |         1         |      172.168.83.12      |
-----------------------------------------------------------------------------
|      172.168.83.12      |         2         |      172.168.83.15      |
-----------------------------------------------------------------------------
|      172.168.83.15      |         3         |      172.168.83.11      |
=============================================================================

                Distribution ID: 3 | State: old | Total segment num: 4

  Primary Segment Node IP        Segment ID        Duplicate Segment node IP
=============================================================================
|      172.168.83.11      |         1         |      172.168.83.12      |
-----------------------------------------------------------------------------
|      172.168.83.12      |         2         |      172.168.83.15      |
-----------------------------------------------------------------------------
|      172.168.83.15      |         3         |      172.168.83.16      |
-----------------------------------------------------------------------------
|      172.168.83.16      |         4         |      172.168.83.11      |
=============================================================================
```

4.5.2.2.1.2Initialize hashmap and perform data redistribution

Operating Steps

Step 1: Initialize hashmap:

```
$ gccli -uroot

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights Reserved.

gbase> use vc vc1;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> initnodedatamap;
Query OK, 0 rows affected, 5 warnings (Elapsed: 00:00:01.45)
```

Step 2: Perform data redistribution:

```
gbase> show variables like '%rebalanc%';
+-----------------------------------------------------------+-----------+
| Variable_ name                                            | Value     |
+-----------------------------------------------------------+-----------+
| _ t_ gcluster_ rebalance_ mirror_ node                    | 0         |
| gcluster_ load_ rebalance_ seed                           | 5         |
| gcluster_ rebalancing_ concurrent_ count                  | 5         |
| gcluster_ rebalancing_ ignore_ mirror                     | OFF       |
| gcluster_ rebalancing_ immediate_ recover_ internal_ table | OFF       |
| gcluster_ rebalancing_ parallel_ degree                   | 4         |
| gcluster_ rebalancing_ random_ table_ quick_ mode         | 1         |
| gcluster_ rebalancing_ step                               | 100000000 |
| gcluster_ rebalancing_ update_ status_ on_ drop_ table    | ON        |
+-----------------------------------------------------------+-----------+
9 rows in set (Elapsed: 00:00:00.24)

gbase> rebalance instance;
Query OK, 2 rows affected (Elapsed: 00:00:01.45)
View Rebalance Status:
gbase> select index_ name, status, percentage  from gclusterdb.rebalancing_ status;
+------------+-----------+------------+
| index_ name | status    | percentage |
+------------+-----------+------------+
| demo.t     | COMPLETED |        100 |
| demo.tt    | COMPLETED |        100 |
+------------+-----------+------------+
2 rows in set (Elapsed: 00:00:00.04)

gbase> quit
Bye
```

4.5.2.2.1.3Delete old distribution

Operating Steps

Step 1: Confirm the current distribution ID. In the current example, the new distribution ID is 4 and the old distribution ID is 3:

```
$ gcadmin showdistribution vc vc1

                Distribution ID: 4 | State: new | Total segment num: 3

  Primary Segment Node IP          Segment ID          Duplicate Segment node IP
=============================================================================
====
|       172.168.83.11       |          1          |       172.168.83.12       |
-----------------------------------------------------------------------------
|       172.168.83.12       |          2          |       172.168.83.15       |
-----------------------------------------------------------------------------
|       172.168.83.15       |          3          |       172.168.83.11       |
=============================================================================
====

                Distribution ID: 3 | State: old | Total segment num: 4

  Primary Segment Node IP          Segment ID          Duplicate Segment node IP
=============================================================================
====
|       172.168.83.11       |          1          |       172.168.83.12       |
-----------------------------------------------------------------------------
|       172.168.83.12       |          2          |       172.168.83.15       |
-----------------------------------------------------------------------------
|       172.168.83.15       |          3          |       172.168.83.16       |
-----------------------------------------------------------------------------
|       172.168.83.16       |          4          |       172.168.83.11       |
=============================================================================
====
```

Step 2: Confirm that there are no tables using the old Distribution ID in the current cluster

```
gbase> select index_ name,tbname,data_ distribution_ id,vc_ id from gbase.table_
distribution;
+------------------------------+------------------+----------------------+---------+
| index_ name                  |tbname            |data_ distribution_ id | vc_ id
|
+------------------------------+------------------+----------------------+---------+
| gclusterdb.rebalancing_ status|rebalancing_ status|                    4 | vc00001 |
| gclusterdb.dual              |dual              |                    4 |
vc00001 |
| demo.t                       |t                 |                    4 |
vc00001 |
| demo.tt                      |tt                |                    4 |
vc00001 |
+------------------------------+------------------+----------------------+---------+
```

Step 3: Delete the old distribution:

```
$ gcadmin rmdistribution 3 vc vc1
cluster distribution ID [3]
it will be removed now
please ensure this is ok, input [Y,y] or [N,n]: y
select count(*) from gbase.nodedatamap where data_ distribution_ id=3 result is not 0
refreshnodedatamap drop 3 success
gcadmin remove distribution [3] success
$ gcadmin showdistribution vc vc1
        Distribution ID: 4 | State: new | Total segment num: 3
 Primary Segment Node IP    Segment ID      Duplicate Segment node IP
=============================================================================
|       172.168.83.11        |        1          |          172.168.83.12          |
-----------------------------------------------------------------
|       172.168.83.12        |        2          |          172.168.83.15          |
-----------------------------------------------------------------
|       172.168.83.15        |        3          |          172.168.83.11          |
-----------------------------------------------------------------
$ gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

==============================================
|        GBASE VIRTUAL CLUSTER INFORMATION        |
==============================================
|      VcName       | DistributionId |    comment    |
----------------------------------------------
|        vc1          |           4            |                |
----------------------------------------------
=============================================================================
|              VIRTUAL CLUSTER DATA NODE INFORMATION                          |
=============================================================================
|NodeName|    IpAddress    |DistributionId| gnode|syncserver|DataState|
-----------------------------------------------------------------
| node1    |172.168.83.11|          4           | OPEN |     OPEN     |      0       |
-----------------------------------------------------------------
| node2    |172.168.83.12|          4           | OPEN |     OPEN     |      0       |
-----------------------------------------------------------------
| node3    |172.168.83.15|          4           | OPEN |     OPEN     |      0       |
-----------------------------------------------------------------
| node4    |172.168.83.16|                      | OPEN |     OPEN     |      0       |
-----------------------------------------------------------------
4 data node
```

4.5.2.2.1.4Removing shrink nodes from the cluster

Operating Steps

Step 1: Copy the gcChangeInfo.xml file to gcRm_ VC1. XML file, and modify the corresponding nodeIP content to the node IP to be modified:

```
$ cp gcChangeInfo.xml gcRm_ vc1.xml
$ vi gcRm_ vc1.xml
$ cat gcRm_ vc1.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
  <rack>
    <node ip="172.168.83.16"/>
  </rack>
</servers>
```

Step 2: Delete the shrink node from the corresponding VC and change it to freenode.

```
$ gcadmin rmnodes gcRm_ vc1.xml vc1
gcadmin remove nodes ...
flush statemachine success
gcadmin rmnodes from vc [vc1] success
$ gcadmin
CLUSTER STATE:                ACTIVE

=================================================================
|                GBASE COORDINATOR CLUSTER INFORMATION                    |
=================================================================
|     NodeName     |      IpAddress      | gcware | gcluster | DataState |
-----------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |         0         |
-----------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |         0         |
-----------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |         0         |
-----------------------------------------------------------------
| coordinator4 | 172.168.83.15 |    OPEN    |     OPEN     |         0         |
-----------------------------------------------------------------
=============================================
|        GBASE VIRTUAL CLUSTER INFORMATION         |
=============================================
|       VcName       | DistributionId |    comment    |
---------------------------------------------
|         vc1          |            4             |                 |
---------------------------------------------
|         vc2          |            2             |                 |
---------------------------------------------
=================================================================
|                GBASE CLUSTER FREE DATA NODE INFORMATION                 |
=================================================================
| NodeName    |      IpAddress      | gnode | syncserver | DataState |
-----------------------------------------------------------------
| FreeNode1 | 172.168.83.16 | OPEN    |       OPEN       |         0         |
-----------------------------------------------------------------
```

4.5.2.2.1.5Uninstall shrink node (optional)

If the node is no longer needed after downsizing, it can be completely removed from the cluster and uninstalled, or it can be used as a freenode for future use.

The steps to completely remove and uninstall shrunk nodes from the cluster are as follows:

Operating Steps

Step 1: Determine gcRm_ The content of the vc1. xml file is the IP address of the node to be uninstalled:

```
$ cat gcRm_ vc1.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
  <rack>
    <node ip="172.168.83.16"/>
  </rack>
</servers>
```

Step 2: Remove the IP to be uninstalled from the cluster

```
$ gcadmin rmnodes gcRm_ vc1.xml
gcadmin remove nodes ...

flush statemachine success

gcadmin rmnodes from cluster success
```

Step 3: Modify the uninstallation configuration file demoUn. options

1) Copy the demo.options file as demoUn.options, and modify the demoUn.options as follows;
2) Set the dataHost parameter to the IP of the node to be uninstalled;
3) Annotate the coordinates Host and coordinates HostNodeID parameters;
4) Modify the existCoordinateHost parameter to the IP address of the coordinator node that will be preserved after resizing;
5) Modify the existDataHost parameter to the IP addresses of all data nodes retained after resizing.

```
$ cat demoUn.options
installPrefix= /opt
#coordinateHost =
#coordinateHostNodeID =
dataHost = 172.168.83.16
existCoordinateHost =172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.15
existDataHost
=172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.14,172.168.83.15
existGcwareHost=172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.15
#gcwareHost =
#gcwareHostNodeID =
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbasedba'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
#characterSet = utf8
#sshPort = 10022
```

Step 4: Stop the cluster service on all data nodes that need to be uninstalled

```
$ gcluster_ services all stop
```

Step 5: Perform uninstallation.

```
$ ./ unInstall.py --silent=demoUn.options
These GCluster nodes will be uninstalled.
CoordinateHost:
DataHost:
172.168.83.16
Are you sure to uninstall GCluster ([Y,y]/[N,n])? y
        172.168.83.16 unInstall 172.168.83.16 successfully.
```

4.5.2.2.2 Pure coordinator node scaling

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13172.168.83.15

Data node:

vc1：172.168.83.11，172.168.83.12

vc2：172.168.83.13，172.168.83.14

Shrink coordinator node IP: 172.168.83.15

4.5.2.2.2.1Preparing configuration files

Operating Steps

Step 1: Modify the demo. options file:

1) Set the coordinatiteHost parameter to the IP of the node to be uninstalled;
2) Set the coordinatiteHostNodeID parameter to the ID set for the node to be uninstalled;
3) Modify the existCoordinateHost parameter to the IP address of the coordinator node that will be preserved after resizing;
4) Modify the existDataHost parameter to the IP addresses of all data nodes retained after resizing.

The modified demo. options refer to the following:

```
$ cat demo.options
installPrefix= /opt
coordinateHost = 172.168.83.15
coordinateHostNodeID =15
#dataHost = 172.168.83.15
existCoordinateHost =172.168.83.11,172.168.83.12,172.168.83.13
existDataHost =172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.14
existGcwareHost=172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.15
#gcwareHost =
#gcwareHostNodeID =
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbasedba'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
```

4.5.2.2.2.2Stop cluster services for all nodes

Operating Steps

Step 1: Execute the cluster service stop command on all nodes.

```
$ gcluster_ services all stop
Stopping gcrecover :                                       [  OK  ]
Stopping gcluster :                                        [  OK  ]
Stopping gbase :                                           [  OK  ]
Stopping syncserver :                                      [  OK  ]
$ gcware_ services all stop
Stopping GCWareMonit success!
Stopping gcware :                                          [  OK  ]
```

4.5.2.2.2.3Uninstalling nodes

Operating Steps

Step 1: Perform uninstallation.

```
$ ./ unInstall.py --silent=demo.options
These GCluster nodes will be uninstalled.
CoordinateHost:
172.168.83.15
DataHost:
Are you sure to uninstall GCluster ([Y,y]/[N,n])? y
        172.168.83.15 unInstall 172.168.83.15 successfully.
Update all coordinator gcware conf.
172.168.83.11 update gcware conf successfully.
172.168.83.12 update gcware conf successfully.
172.168.83.13 update gcware conf successfully.
```

4.5.2.2.2.4Start cluster services for all nodes

Operating Steps

Step 1: Execute the cluster service startup command on all nodes.

```
$ gcluster_ services all start
Starting gbase :                                           [   OK   ]
Starting syncserver :                                      [   OK   ]
Starting gcluster :                                        [   OK   ]
Starting gcrecover :                                       [   OK   ]
$ gcware_ services all start
Starting gcware :                                          [   OK   ]
Starting GCWareMonit success!
The status information after installation is as follows:
$ gcadmin
CLUSTER STATE:              ACTIVE


=============================================================
|                GBASE COORDINATOR CLUSTER INFORMATION                |
=============================================================
|    NodeName    |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
```

```
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------------
===================================================================
|                GBASE CLUSTER FREE DATA NODE INFORMATION                |
===================================================================
| NodeName    |     IpAddress      | gnode | syncserver | DataState |
-------------------------------------------------------------------
| FreeNode1 | 172.168.83.11 | OPEN    |      OPEN      |        0        |
-------------------------------------------------------------------
| FreeNode2 | 172.168.83.12 | OPEN    |      OPEN      |        0        |
-------------------------------------------------------------------
| FreeNode3 | 172.168.83.13 | OPEN    |      OPEN      |        0        |
-------------------------------------------------------------------
| FreeNode4 | 172.168.83.14 | OPEN    |      OPEN      |        0        |
-------------------------------------------------------------------

0 virtual cluster
3 coordinator node
4 free data node
```

4.5.2.2.3 Composite node scaling

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13172.168.83.15

Data node:

vc1：172.168.83.11，172.168.83.12，172.168.83.15

vc2：172.168.83.13，172.168.83.14

Node IP to be scaled down: 172.168.83.15

4.5.2.2.3.1Create a new distribution

Operating Steps

Step 1: Copy gcChangeInfo.xml from the installation directory to gcS_ VC1. XML file, and then remove the node IP to be shrunk.

Modified gcS_ The vc1. XML file reference is as follows:

```
$cp gcChangeInfo.xml gcS_ vc1.xml
$cat gcS_ vc1.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
 <rack>
  <node ip="172.168.83.11"/>
  <node ip="172.168.83.12"/>
 </rack>
</servers>
```

Step 2: Execute the command to create a distribution.

```
$ gcadmin distribution gcS_ vc1.xml p 1 d 1 vc vc1
gcadmin generate distribution ...
gcadmin generate distribution successful
The completed cluster information is as follows:
$ gcadmin
CLUSTER STATE:              ACTIVE

=================================================================
|                GBASE COORDINATOR CLUSTER INFORMATION                |
=================================================================
|    NodeName    |     IpAddress     | gcware | gcluster | DataState |
-----------------------------------------------------------------
| coordinator1 | 172.168.83.11 |   OPEN   |    OPEN    |       0       |
-----------------------------------------------------------------
| coordinator2 | 172.168.83.12 |   OPEN   |    OPEN    |       0       |
-----------------------------------------------------------------
| coordinator3 | 172.168.83.13 |   OPEN   |    OPEN    |       0       |
-----------------------------------------------------------------
| coordinator4 | 172.168.83.15 |   OPEN   |    OPEN    |       0       |
-----------------------------------------------------------------

=============================================
|       GBASE VIRTUAL CLUSTER INFORMATION       |
=============================================
|      VcName      | DistributionId |   comment   |
---------------------------------------------
|        vc1         |         4,5          |               |
---------------------------------------------
|        vc2         |          2           |               |
---------------------------------------------

2 virtual cluster: vc1, vc2
4 coordinator node
0 free data node
```

```
$ gcadmin distribution gcS_ vc1.xml p 1 d 1 vc vc1
gcadmin generate distribution ...
gcadmin generate distribution successful
$ gcadmin showdistribution vc vc1

      Distribution ID: 5 | State: new | Total segment num: 2

   Primary Segment Node IP     Segment ID     uplicate Segment node IP
==========================================================================
|        172.168.83.11        |       1       |        172.168.83.12        |
--------------------------------------------------------------------------
|        172.168.83.12        |       2       |        172.168.83.11        |
==========================================================================

      Distribution ID: 4 | State: old | Total segment num: 3


   Primary Segment Node IP     Segment ID      Duplicate Segment node IP
============================================================================
|        172.168.83.11        |       1       |        172.168.83.12        |
----------------------------------------------------------------------------
|        172.168.83.12        |       2       |        172.168.83.15        |
----------------------------------------------------------------------------
|        172.168.83.15        |       3       |        172.168.83.11        |
============================================================================
```

4.5.2.2.3.2Initialize hashmap and perform data redistribution

Operating Steps

Step 1: Initialize hashmap:

```
$ gccli -uroot


GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.    All Rights Reserved.


gbase> use vc vc1;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> initnodedatamap;
Query OK, 0 rows affected, 5 warnings (Elapsed: 00:00:01.45)
```

Step 2: Perform data redistribution:

```
gbase> rebalance instance;
Query OK, 3 rows affected (Elapsed: 00:00:01.45)
View Rebalance Status:
gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_ status;
+------------+-----------+------------+
```

```
gbase> rebalance instance;
Query OK, 3 rows affected (Elapsed: 00:00:01.45)
| index_ name | status        | percentage |
+------------+-----------+------------+
| demo.tt        | COMPLETED |              100 |
| demo.ttt      | COMPLETED |              100 |
| demo.t         | COMPLETED |              100 |
+------------+-----------+------------+
3 rows in set (Elapsed: 00:00:00.05)

gbase> quit
Bye
```

4.5.2.2.3.3Delete old distribution

Operating Steps

Step 1: Confirm the current distribution ID. In the current example, the new distribution ID is 5 and the old distribution ID is 4:

```
$ gcadmin showdistribution vc vc1

      Distribution ID: 5 | State: new | Total segment num: 2

   Primary Segment Node IP     Segment ID     uplicate Segment node IP
=============================================================================
|         172.168.83.11          |        1         |         172.168.83.12          |
-----------------------------------------------------------------------
|         172.168.83.12          |        2         |         172.168.83.11          |
=============================================================================

      Distribution ID: 4 | State: old | Total segment num: 3

   Primary Segment Node IP     Segment ID      Duplicate Segment node IP
=============================================================================
|         172.168.83.11          |        1         |           172.168.83.12          |
-----------------------------------------------------------------------
|         172.168.83.12          |        2         |           172.168.83.15          |
-----------------------------------------------------------------------
|         172.168.83.15          |        3         |           172.168.83.11          |
=============================================================================
```

Step 2: Confirm that there are no tables using the old Distribution ID in the current cluster

```
gbase> select index_ name,tbname,data_ distribution_ id,vc_ id from gbase.table_
distribution where vc_ id='vc00001';
+----------------------------+-----------------+--------------------+-------+
|index_ name                 |tbname           |data_ distribution_ id|vc_
id    |
+----------------------------+-----------------+--------------------+-------+
|gclusterdb.rebalancing_ status|rebalancing_ status|                  5|vc00001|
|demo.ttt                    |ttt              |
5|vc00001|
|gclusterdb.dual             |dual             |
5|vc00001|
|demo.t                      |t                |
5|vc00001|
|demo.tt                     |tt               |
5|vc00001|
+----------------------------+-----------------+--------------------+-------+
5 rows in set (Elapsed: 00:00:00.00)
```

Step 3: Delete the old distribution:

```
$ gcadmin rmdistribution 4 vc vc1
cluster distribution ID [4]
it will be removed now
please ensure this is ok, input [Y,y] or [N,n]: y
select count(*) from gbase.nodedatamap where data_ distribution_ id=4 result is not 0
refreshnodedatamap drop 4 success
gcadmin remove distribution [4] success
$ gcadmin showdistribution vc vc1

      Distribution ID: 5 | State: new | Total segment num: 2

  Primary Segment Node IP     Segment ID     uplicate Segment node IP
=============================================================================
|         172.168.83.11          |        1         |          172.168.83.12          |
-----------------------------------------------------------------------
|         172.168.83.12          |        2         |          172.168.83.11          |
=============================================================================

$ gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

============================================
|        GBASE VIRTUAL CLUSTER INFORMATION        |
============================================
|      VcName       | DistributionId |    comment    |
--------------------------------------------
|        vc1          |           5            |                |
--------------------------------------------
=================================================================
|              VIRTUAL CLUSTER DATA NODE INFORMATION                          |
=================================================================
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-----------------------------------------------------------------
| node1    |172.168.83.11|          5           |OPEN |     OPEN     |      0      |
-----------------------------------------------------------------
| node2    |172.168.83.12|          5           |OPEN |     OPEN     |      0      |
-----------------------------------------------------------------
| node3    |172.168.83.15|                      |OPEN |     OPEN     |      0      |
-----------------------------------------------------------------

3 data node
```

4.5.2.2.3.4Removing shrink nodes from the cluster

Operating Steps

Step 1: Modify gcRm_ VC1. XML, where the corresponding nodeIP content is the node IP to be modified:

```
$ cp gcChangeInfo.xml gcRm_ vc1.xml
$ vi gcRm_ vc1.xml
$ cat gcRm_ vc1.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
  <rack>
    <node ip="172.168.83.15"/>
  </rack>
</servers>
```

Step 2: Remove the shrink node from the corresponding VC

```
$ gcadmin rmnodes gcRm_ vc1.xml vc1
gcadmin remove nodes ...

flush statemachine success

gcadmin rmnodes from vc [vc1] success

$ gcadmin
CLUSTER STATE:                ACTIVE

=================================================================
|                  GBASE COORDINATOR CLUSTER INFORMATION                    |
=================================================================
|     NodeName     |     IpAddress     | gcware | gcluster | DataState |
-----------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |         0         |
-----------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |         0         |
-----------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |         0         |
-----------------------------------------------------------------
| coordinator4 | 172.168.83.15 |    OPEN    |     OPEN     |         0         |
-----------------------------------------------------------------
=============================================
|        GBASE VIRTUAL CLUSTER INFORMATION        |
=============================================
|       VcName       | DistributionId |    comment    |
---------------------------------------------
|         vc1          |            5             |                  |
---------------------------------------------
|         vc2          |            2             |                  |
---------------------------------------------
=================================================================
|                GBASE CLUSTER FREE DATA NODE INFORMATION                  |
=================================================================
| NodeName    |     IpAddress     | gnode | syncserver | DataState |
-----------------------------------------------------------------
| FreeNode1 | 172.168.83.15 | OPEN    |       OPEN       |         0         |
-----------------------------------------------------------------
```

4.5.2.2.3.5Stop cluster services for all nodes

Operating Steps

Step 1: Execute the cluster service stop command on all nodes.

```
$ gcluster_ services all stop
Stopping gcrecover :                                        [   OK   ]
Stopping gcluster :                                         [   OK   ]
```

```
Stopping gbase :                                           [  OK  ]
Stopping syncserver :                                      [  OK  ]
$ gcware_ services all stop
Stopping GCWareMonit success!
Stopping gcware :                                          [  OK  ]
```

4.5.2.2.3.6Uninstalling nodes

Operating Steps

Step 1: Modify the configuration file of the uninstallation node.

1) Set the coordinatiteHost parameter to the IP of the node to be uninstalled;
2) Set the coordinatiteHostNodeID parameter to the ID set for the node to be uninstalled;
3) Set the dataHost parameter to the IP of the node to be uninstalled;
4) Modify the existCoordinateHost parameter to the IP address of the coordinator node that will be preserved after resizing;
5) Modify the existDataHost parameter to the IP addresses of all data nodes retained after resizing.

```
$ cat demoUn.options
installPrefix= /opt
coordinateHost =172.168.83.15
coordinateHostNodeID =15
dataHost =172.168.83.15
existCoordinateHost =172.168.83.11,172.168.83.12,172.168.83.13
existDataHost =172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.14
existGcwareHost=172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.15
#gcwareHost =
#gcwareHostNodeID =
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbasedba'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
```

Step 4: Perform uninstallation.

```
$ ./ unInstall.py --silent=demoUn.options
These GCluster nodes will be uninstalled.
CoordinateHost:
172.168.83.15
DataHost:
172.168.83.15
Are you sure to uninstall GCluster ([Y,y]/[N,n])? y
        172.168.83.15 unInstall 172.168.83.15 successfully.
Update all coordinator gcware conf.
172.168.83.11 update gcware conf successfully.
172.168.83.12 update gcware conf successfully.
172.168.83.13 update gcware conf successfully.
```

4.5.2.2.3.7Start cluster services for all nodes

Operating Steps

Step 1: Execute the cluster service startup command on all nodes.

```
$ gcluster_ services all start
Starting gbase :                                           [  OK  ]
Starting syncserver :                                      [  OK  ]
Starting gcluster :                                        [  OK  ]
Starting gcrecover :                                       [  OK  ]
$ gcware_ services all start
Starting gcware :                                          [  OK  ]
Starting GCWareMonit success!

The status information after installation is as follows:
$ gcadmin
CLUSTER STATE:              ACTIVE

=============================================================
|                GBASE COORDINATOR CLUSTER INFORMATION                |
=============================================================
|    NodeName    |    IpAddress    | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
=============================================
|       GBASE VIRTUAL CLUSTER INFORMATION       |
=============================================
```

```
|      VcName      | DistributionId |   comment   |
-----------------------------------------------
|       vc1        |        5       |             |
-----------------------------------------------
|       vc2        |        2       |             |
-----------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node
```

4.5.3 Cluster node replacement

As the cluster size continues to expand, the probability of node damage in the cluster increases. At the same time, as the amount of data increases, the computing and storage capabilities of individual nodes may also become bottlenecks. In both cases, it is necessary to replace the cluster nodes to ensure that the cluster can work normally. GBase 8a MPP Cluster has the ability to provide online non-stop service for node replacement, allowing for node replacement without downtime in production environments.

explain

- Within the cluster, only one node replacement process can be started at the same time, and multiple node replacement processes are not allowed to run in parallel. GCware will add resource locks to ensure this.
- Only nodes in the same VC can be replaced at once, and the replaced node must have at least one available backup shard:
  - Pure data nodes, within a VC, can replace multiple data nodes at once;
  - Composite nodes require two replacements, first replacing the coordinator node and then replacing the data node;
  - Composite nodes can simultaneously replace multiple coordinator nodes, including the coordinator node in the composite node;
  - Composite nodes can simultaneously replace multiple data nodes in a VC, including the data nodes in the composite node.
  - If gcware is deployed on the composite node, it needs to be replaced three times: first perform gcware replacement, then coordinator replacement, and finally perform data node replacement.

- Allow interruption of executing node replacement processes.
- Allow users to specify the recovery order of user data tables during data recovery.
- Allow users to specify whether to use memory (/dev/shm) to store temporary data during node replacement.
- Instructions for replacing online non-stop service nodes:

  When a node is replaced, the cluster changes to a readonly state. When the cluster receives an application write operation, it allows SQL to wait without returning an error message. After the node replacement operation is completed, it returns. If the application does not time out, it can continue to execute to ensure that the node is completely online during replacement.

- If gcware and gccluster are deployed on the same node, they need to be replaced separately during node replacement. First, replace the gcware node, and then replace the gccluster node.
  - The GCluster node replacement operation is the same as the old version node replacement operation, without any changes.
  - The replacement of gcware nodes needs to be done separately, and before replacement, it must be ensured that the majority of gcware nodes are available.
  - It can replace less than half of the total number of gcware nodes at once.
  - Supports online replacement without the need to set the gcware status of the original cluster gcware node.

4.5.3.1 Node Replacement Command

This section mainly describes the parameter descriptions of the node replacement command replace.py and the stop replacement command replaceStop.py. Please refer to sections 4.5.3.2 to 4.5.3.5 for specific replacement steps as needed.

4.5.3.1.1 Replace Installation Command

Use the replace. py command to replace and install the node to be replaced. The cluster installation user dbauser must be used on the coordinator node in the cluster to execute commands.

Command format:

```
replace.py [options]
```

Table -439 Parameter Description

Parameter Name	explain
-h，--help	Display help information for replace. py.
-a	Do not prompt the user for confirmation information.
---host=HOSTLIST	Specify the list of node IPs to be replaced, separated by commas.
--type=NODETYPE	Node type being replaced: Coor: Replace the management node; Data: Replace the data node.
--freenode=FREENODE	Specify the idle data node IP to be replaced, which corresponds one-to-one with the IP specified in the --host. When replacing pure data nodes, replace damaged pure data nodes with idle data nodes that are not used by any virtual cluster.
--dbaUser	The operating system DBA username used during cluster runtime.
--dbaUserPwd	The password of the operating system DBA user must be consistent across all nodes.
--generalDBUser	Database DBA username, optional parameter, defaults to database root user when not entered.
--generalDBPwd	Database DBA user password. Currently, single quotes are not supported in passwords, and other special symbols are surrounded by single quotes.
--overwrite	Force replacement flag. If this flag is set, the remaining cluster software on the replaced node will be forcibly uninstalled and a new cluster will be reinstalled. This parameter is optional.
--sync_ coordi_ metadata_ timeout	The timeout time, in minutes, for each copy of coordinator node metadata when performing node replacement. The default is 15 minutes, with a minimum value of 1 and a maximum value of 2147483647.
--parallel_ pack	The backup copy mode of coordinator node metadata, with values of 0 or 1, defaults to 0. 0- Full backup of each node; 1- Each node backs up a portion and finally summarizes it at the destination node.
--retry_ times	The number of retries that a single step operation fails during the replacement process, with a default value of 3, a minimum value of 1, and a maximum value of 2147483647.
--use_ shm	Use shared memory to store data packets. To use this parameter, first confirm that/dev/shm of each

Parameter Name	explain
	coordinator node has enough space to store all cluster layer metadata. The value is 0 or 1, and the default value is 0.
--vcname	The VC name of the replaced data node can only replace one VC at a time.
--passwordputMode	Used to specify the method of password acquisition, and achieve different acquisition methods through different parameters. If this parameter is specified, the password in demo. options does not need to be modified again. The value range is file, pwdsame, pwddiff, and the default value is file. The value description is as follows: ● File: represents obtaining from a file or command line parameter, consistent with the original method, in which the password in the file is plaintext; ● Pwdname: This parameter is used when the password is entered by the user from the terminal and the passwords of all nodes are consistent. For different user passwords, only one input is required; ● Pwddiff:: This parameter is used when the password is entered by the user from the terminal and the passwords between nodes are inconsistent. For different user passwords, each node is input once.

 **be careful**

- Node replacement related log description:

  Replace.log records the detailed gcadmin call operations performed during the installation and uninstallation of the replaced node, as well as the check information and ssh interaction information of logging in each node;

  Gcware node gcware/liblog/lower gadm_ cp_ codi_ Tbl. log and gcware/gadm_ cp_ sys_ The synchronization metadata and data related information are recorded in tbl. log.

4.5.3.1.2 Stop Replace Installation Command

Function Description

Node replacement can be stopped during the execution process by executing replaceStop.py. If the stopped node replaces multiple nodes simultaneously, the replacement process of all replaced nodes will be stopped.

 **explain**

- The cluster has a node replacement operation running.
- Execute the replaceStop.py script using a DBA user on a normally available Coordinator node.
- If the recovery cluster mode is NORMAL and the node state cannot be rolled back, the node state may be UNAVAILABLE or REPLACE.

 **warning**

During the node replacement process, it is only necessary to run the replaceStop.py script to stop the node replacement operation if there are some software and hardware exceptions, or if the node replacement operation cannot be completed for a long time. Please note that you cannot manually kill node replacement related processes to stop node replacement.

Grammar format

```
replaceStop.py --host=<HOST_ LIST> --type=<type_ value> --dbaUser=<username> --dbaUserPwd=<password>
```

Table 440 Parameter Description

Parameter Name	Description
HOST_ LIST	Specify the IP list of nodes to be stopped, which is the same as the specified node list when executing the node replacement command, separated by commas.
type_ value	The type of node being stopped --The node whose type=data has been stopped for replacement is a data node; --The node whose type=coor has been stopped and replaced is the management node.
username	The operating system username used during cluster runtime.
password	The password of the operating system user must be consistent across all nodes.

Example

Coordinator node replacement process stopped:

- On the node where the replacement command is executed, execute the following command with a DBA user to start the coordinator node replacement;

```
./replace.py --host=192.168.6.105 --type=coor --dbaUser=gbase --dbaUserPwd=gbase –overwrite --vcname=vc1
```

- When the "build data packet start" message is printed, it indicates that packaging is in progress. At this time, execute the command on the node to stop replacing the coordinator node.

```
$ ./ replaceStop.py --host=192.168.6.105 --type=coor --dbaUser=gbase --dbaUserPwd=gbase
Checking environment…
Stop python replace.py on host 192.168.6.101.
Stop install python scripts on host 192.168.6.105.
Stop gcadmin replacenodes on host 192.168.6.101.
Stop metadata sync on host 192.168.6.101, 192.168.6.102, 192.168.6.103, 192.168.6.104, 192.168.6.105.
Clean up temporary on coordinators 192.168.6.101, 192.168.6.102, 192.168.6.103, 192.168.6.104, 192.168.6.105.
Clean up temporary ondatanodes 192.168.6.101
Stop gcluster nodes 192.168.6.105.
```

4.5.3.2 Replacement of gcware nodes

GCware nodes need to be replaced separately during node replacement:

- Deploying gcware and gcluster on the same server for node replacement

  The replacement requires two steps, replacing gccluster and gcware separately. It is recommended to replace gcware first and then proceed with gccluster replacement after success.

  If there is still gnode on the server, replace it with gnode at the end.

  The GCluster node replacement operation is the same as the old version node replacement operation, without any changes.

  The replacement of gcware nodes needs to be done separately, following the steps for replacing gcware nodes.

- Independent deployment of gcware on one server for node replacement

  Deploy gcware nodes independently for node replacement, following the steps for gcware node replacement.

Steps for replacing gcware nodes:

Before replacing, it is necessary to ensure that the majority of gcware nodes (more than half of the total number of gcware nodes) are available, and less than half of the total number of gcware nodes can be replaced at once. Supports online replacement without the need to set the

gcware status of the original cluster gcware node. The specific operation is as follows:

1. Prepare a new node with the same system environment and IP as the problem node
2. There is an installation directory for the cluster in the new node (installPrefix in demo. option) with the same permissions as the problem node
3. Perform a replacement on a node with normal gcware service, as shown in the following example:

```
cd $GCWARE_ BASE/gcware_ server/
./gcserver.py --prefix=/opt --host=192.168.146.21 --dbaUser=gbase --dbaPwd=gbase --overwrite
```

The syntax for replacing GCware nodes is as follows:

```
gcserver.py [options]
Options:
-h. -- help
-A Mask command interaction
--Host=GCWAREHOST The gcware node that needs to be replaced
--DbaUser=DBAUSER corresponds to the dbaUser parameter of demo. options, default gbase
--DbaPwd=dbaPwd corresponding to demo. options in DBAPWD, default gbase
  --overwrite new and complete overwrite
```

be careful

- It can replace less than half of the total number of gcware nodes at once.
- Supports online replacement without the need to set the gcware status of the original cluster gcware node.
- Node replacement log in current execution directory: $GCWARE_ BASE/gcware_ server/gcware_ replace.log

4.5.3.3 Replacement of Pure Data Nodes

When replacing nodes with pure data nodes, the freenode in the cluster can be directly used. If there is no freenode, a new machine outside the cluster can also be used for node replacement.

The comparison is as follows:

Table -441 Comparison between Freenode Replacement and New Machine Replacement

Replacement steps	Freenode	New machine
Before replacement	The cluster software has been installed and freenode can be seen in the cluster	1、The new machine needs to install the operating system of the original machine system and meet the cluster installation requirements 2、Set the new machine IP to be consistent with the replaced node IP 3、No cluster software installed
After replacement	The IP of cluster nodes will change	The IP of the cluster remains unchanged

4.5.3.3.1 Freenode replaces a pure Data node

Replacement instructions

- Create a new distribution and redistribute data from the old distribution to the newly created distribution after successful node replacement, with the sharding on non replacement nodes unchanged.
- When replacing a pure data node of a VC, it can access other VC's

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13

Data node:

vc1：172.168.83.11，172.168.83.12，172.168.83.15

Vc2：172.168.83.13，172.168.83.14

Freenode：172.168.83.16

Replace 172.168.83.15 with 172.168.83.16.

4.5.3.3.1.1Installation node (optional)

If a freenode already exists in the cluster, you can skip this section.

Operating Steps

Step 1: Modify the demo. options file:

1) Set the dataHost parameter to the IP of the node to be installed;
2) Modify the existCoordinateHost parameter to the IP of an existing coordinator node;
3) Modify the existDataHost parameter to the IP addresses of all existing data nodes.

The modified demo. options refer to the following:

```
$ cat demo.options
installPrefix= /opt
#coordinateHost =172.168.83.11,172.168.83.12,172.168.83.13
#coordinateHostNodeID = 11,12,13
dataHost =172.168.83.16
existCoordinateHost =172.168.83.11,172.168.83.12,172.168.83.13
existDataHost
=172.168.83.11,172.168.83.12,172.168.83.13,172.168.83.14,172.168.83.15
dbaUser = gbase
dbaGroup = gbase
dbaPwd = 'gbasedba'
rootPwd = '111111'
#rootPwdFile = rootPwd.json
```

Step 2: Perform the installation.

```
$ ./ gcinstall.py --silent=demo.options
************************************************************************
***************
Thank you for choosing GBase product!


……


************************************************************************
Do you accept the above licence agreement ([Y,y]/[N,n])? y
************************************************************************
                        Welcome to install GBase products
************************************************************************
Environmental Checking on gcluster nodes.
CoordinateHost:
DataHost:
172.168.83.16
Are you sure to install GCluster on these nodes ([Y,y]/[N,n])? y


……


172.168.83.16                install cluster on host 172.168.83.16 successfully.
update and sync configuration file...
Starting all gcluster nodes...
adding new datanodes to gcware...
$

##The above message indicates successful installation
The status information after installation is as follows:
$ gcadmin
```

```
CLUSTER STATE:                ACTIVE


=============================================================
=====
|                          GBASE  COORDINATOR  CLUSTER  INFORMATION
|
=============================================================
=====
|     NodeName     |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
=========================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                |
=========================================================
|       VcName       | DistributionId |            comment            |
---------------------------------------------------------
|         vc1          |             1             | comment message vc1 |
---------------------------------------------------------
|         vc2          |             2             | comment message vc2 |
---------------------------------------------------------
=============================================================
===
|                       GBASE  CLUSTER  FREE  DATA  NODE  INFORMATION
|
=============================================================
===
| NodeName    |     IpAddress     | gnode | syncserver | DataState |
-------------------------------------------------------------
| FreeNode1 | 172.168.83.16 | OPEN    |       OPEN       |        0        |
-------------------------------------------------------------


2 virtual cluster: vc1, vc2
3 coordinator node
1 free data node
```

4.5.3.3.1.2Set node status and clean feventlog

Check the node status, the cluster status should be normal, and the coordinator node status should be normal. After confirming that the replaced node is a pure Data node, set the replaced node status to unavailable.

Operating Steps

Step 1: Check the node status. The cluster status should be normal, the coordinator node status should be normal, and the replaced node should be DATA NODE.

```
$gcadmin showcluster vc vc1
CLUSTER STATE:              ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL


=============================================================
|              GBASE VIRTUAL CLUSTER INFORMATION              |
=============================================================
|      VcName      | DistributionId |           comment           |
-------------------------------------------------------------
|        vc1          |           1             | comment message vc1 |
-------------------------------------------------------------
=============================================================================
==========
|                                VIRTUAL CLUSTER DATA NODE INFORMATION
|
=============================================================================
==========
| NodeName |     IpAddress        | DistributionId |      gnode      |syncserver|DataState |
-----------------------------------------------------------------------------
|   node1    | 172.168.83.11    |          1           |      OPEN       |    OPEN    |      0       |
-----------------------------------------------------------------------------
|   node2    | 172.168.83.12    |          1           |      OPEN       |    OPEN    |      0       |
-----------------------------------------------------------------------------
|   node3    | 172.168.83.15    |          1           |      OPEN       |    OPEN    |      0       |
-----------------------------------------------------------------------------


3 data node
```

Step 2: Set the status of the replaced node to unavailable: Run the gcadmin setnodestate command under the DBA user (specified by the dbauser parameter in the demo. options file) of the operating system to set the status of the node to be replaced to unavailable.

```
$ gcadmin setnodestate 172.168.83.15 unavailable
after set node state into unavailable,can not set the state into normal,
must run gcadmin replacenodes to replace this node ,after that command node state can return into
normal.
you realy want to set node state into unavailable(yes or no)?
yes
get node data state by ddl fevent log start ......
get node data state by ddl fevent log end ......
get node data state by dml fevent log start ......
get node data state by dml fevent log end ......
```

```
get node data state by dml storage fevent log start ......
get node data state by dml storage fevent log end ......

check data server node data state by fevent log start ......
check data server node data state by fevent log end ......
set node [172.168.83.15] state to unavailable successful
View cluster status:
$gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

=========================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                |
=========================================================
|       VcName       | DistributionId |            comment            |
---------------------------------------------------------
|         vc1          |            1             | comment message vc1 |
---------------------------------------------------------
=============================================================================
==========
|                                  VIRTUAL CLUSTER DATA NODE INFORMATION
|
=============================================================================
==========
| NodeName |      IpAddress         | DistributionId |       gnode       |syncserver|DataState |
-----------------------------------------------------------------------------
|    node1     | 172.168.83.11     |            1             |       OPEN        |     OPEN     |       0        |
-----------------------------------------------------------------------------
|    node2     | 172.168.83.12     |            1             |       OPEN        |     OPEN     |       0        |
-----------------------------------------------------------------------------
|    node3     | 172.168.83.15     |            1             | UNAVAILABLE |              |                |
-----------------------------------------------------------------------------

3 data node
```

Step 3: Delete the feventlog of the replaced node.

```
$ gcadmin rmfeventlog 172.168.83.15
after rmfeventlog 172.168.83.15, fevent log will be removed, must run gcadmin
replacenodes to replace this node.
you realy want to remove node 172.168.83.15 fevent log(yes or no)?
yes
delete ddl event log on node 172.168.83.15 start
delete ddl event log on node 172.168.83.15 end
delete dml event log on node 172.168.83.15 start
delete dml event log on node 172.168.83.15 end
```

```
delete dml storage event log on node 172.168.83.15 start
delete dml storage event log on node 172.168.83.15 end
```

4.5.3.3.1.3Create an intermediate distribution

Establish a new distribution, which is used to eliminate replaced nodes while maintaining the sharded distribution of other nodes.

Operating Steps

Step 1: View the distribution information of vc1 where node 172.168.83.15 is located.

```
$ gcadmin showdistribution vc vc1 node
        Distribution ID: 1 | State: new | Total segment num: 3

============================================================
====
|   nodes     |    172.168.83.11    | 172.168.83.12 | 172.168.83.15    |
------------------------------------------------------------
| primary    |            1              |          2             |          3             |
| segments |                             |                        |                        |
------------------------------------------------------------
|duplicate |            3              |          1             |          2             |
|segments 1|                             |                        |                        |
============================================================
====
```

Step 2: Use the gcadmin getdistribution command to save the distribution information of the VC where the node to be replaced is located in the specified file.

From the execution result of step 1, it can be seen that the Distribution ID is 1. Save the distribution information with Distribution ID 1 on vc1 to the file Distribution_ info_ In vc1. xml:

```
$ gcadmin getdistribution 1 distribution_ info_ vc1.xml vc vc1
gcadmin getdistribution 1 distribution_ info_ vc1.xml vc vc1 ...

get segments information
write segments information to file [distribution_info_vc1.xml]

gcadmin getdistribution information successful

$ cat distribution_ info_ vc1.xml
<? xml version='1.0' encoding="utf-8"?>
<distributions>
    <distribution>
        <segments>
            <segment>
                <primarynode ip="172.168.83.11"/>
```

```
                <duplicatenodes>
                    <duplicatenode ip="172.168.83.12"/>
                </duplicatenodes>
            </segment>

            <segment>
                <primarynode ip="172.168.83.12"/>

                <duplicatenodes>
                    <duplicatenode ip="172.168.83.15"/>
                </duplicatenodes>
            </segment>

            <segment>
                <primarynode ip="172.168.83.15"/>

                <duplicatenodes>
                    <duplicatenode ip="172.168.83.11"/>
                </duplicatenodes>
            </segment>
        </segments>
    </distribution>
</distributions>
```

Step 3: Modify the distribution rule information for the new distribution.

The modification principle is to ensure that the replaced node has no sharding, and the distribution rules of sharding for other nodes remain unchanged,

- If the shard stored by the replaced node is used as the primary shard, modify the backup shard node of the shard to the primary shard node, that is, if the node IP is in the primary node label, replace the IP in the duplicate nodes label of the segment with the IP of the replaced node, and delete the duplicate nodes label.
- If the shards stored by the replaced node are used as backup shards, that is, if the IP of the replaced node is in the duplicate nodes label, the duplicate nodes label will be deleted.

Modified distribution_ info_ The vc1. XML file reference is as follows:

```
$ cat distribution_ info_ vc1.xml
<? xml version='1.0' encoding="utf-8"?>
<distributions>
    <distribution>
        <segments>
            <segment>
                <primarynode ip="172.168.83.11"/>
```

```
                <duplicatenodes>
                    <duplicatenode ip="172.168.83.12"/>
                </duplicatenodes>
            </segment>

            <segment>
                <primarynode ip="172.168.83.12"/>
            </segment>

            <segment>
                <primarynode ip="172.168.83.11"/>
            </segment>
        </segments>
    </distribution>
</distributions>
```

Step 4: Modify the gcChangeInfo required to create a distribution_ VC1. XML file.

```
$ cat gcChangeInfo_ vc1.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
  <cfgFile file="distribution_info_vc1.xml"/>
</servers>
```

Step 5: Execute the creation of a new distribution with a distribution ID of 3.

```
$ gcadmin distribution gcChangeInfo_ vc1.xml vc vc1
gcadmin generate distribution ...


gcadmin generate distribution successful
The completed cluster information is as follows:
$ gcadmin showdistribution vc vc1


        Distribution ID: 3 | State: new | Total segment num: 3


  Primary Segment Node IP    Segment ID     Duplicate Segment node IP
=============================================================
=======
|        172.168.83.11          |      1         |          172.168.83.12            |
```

```
-------------------------------------------------------------
|       172.168.83.12          |      2          |                                  |
-------------------------------------------------------------
|       172.168.83.11          |      3          |                                  |
=============================================================
=======

          Distribution ID: 1 | State: old | Total segment num: 3

   Primary Segment Node IP Segment ID      Duplicate Segment node IP
=============================================================
=======
|          172.168.83.11         |      1          |          172.168.83.12           |
-------------------------------------------------------------
|          172.168.83.12         |      2          |          172.168.83.15           |
-------------------------------------------------------------
|          172.168.83.15         |      3          |          172.168.83.11           |
=============================================================
=======
```

$ gcadmin showdistribution vc vc1 node

```
          Distribution ID: 3 | State: new | Total segment num: 3

=============================================================
======
|   nodes     |   172.168.83.11   |           172.168.83.12                  |
-------------------------------------------------------------
| primary   |           1              |                  2                            |
| segments |           3              |                                                |
-------------------------------------------------------------
|duplicate |                          |                  1                            |
|segments 1|                          |                                                |
=============================================================
======

          Distribution ID: 1 | State: old | Total segment num: 3

=============================================================
```

```
======
|   nodes    |   172.168.83.11 | 172.168.83.12   | 172.168.83.15      |
----------------------------------------------------------------
| primary   |          1          |         2          |         3          |
| segments |                      |                    |                    |
----------------------------------------------------------------
|duplicate |          3          |         1          |         2          |
|segments 1|                      |                    |                    |
================================================================
======
```

4.5.3.3.1.4Initialize hashmap and perform data redistribution

Execute the initnodeDatamap command to initialize the hashmap, and then redistribute the data to the latest distribution (Distribution ID: 2) through the rebalance instance command.

 **explain**

- According to the distribution rules, this rebalance operation will not actually perform data movement, so it will be completed quickly;
- Do not delete the old version of nodeDatamap and distribution after this rebalance operation.

Operating Steps

Step 1:

Step 1: Initialize hashmap:

```
$ gccli -uroot
GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.   All Rights Reserved.


gbase> use vc vc1;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> initnodedatamap;
Query OK, 0 rows affected, 5 warnings (Elapsed: 00:00:01.45)
```

Step 2: Perform data redistribution:

```
gbase> rebalance database demo;
Query OK, 2 rows affected (Elapsed: 00:00:01.45)
View Rebalance Status:
gbase> rebalance instance;
Query OK, 3 rows affected (Elapsed: 00:00:05.60)


gbase>  select  index_  name,status,percentage,priority,host,distribution_  id  from
```

```
gbase> rebalance database demo;
Query OK, 2 rows affected (Elapsed: 00:00:01.45)
gclusterdb.rebalancing_ status;
+------------+-----------+------------+----------+---------------+-----------------+
| index_ name | status       | percentage | priority | host              | distribution_ id |
+------------+-----------+------------+----------+---------------+-----------------+
| demo.tt       | COMPLETED |              100 |              5 | 172.168.83.11 |                        3 |
| demo.t         | COMPLETED |              100 |              5 | 172.168.83.11 |                        3 |
| demo.ttt      | COMPLETED |              100 |              5 | 172.168.83.11 |                        3 |
+------------+-----------+------------+----------+---------------+-----------------+
3 rows in set (Elapsed: 00:00:00.17)

gbase> quit
Bye
```

4.5.3.3.1.5Execute node replacement command

Function Description

Replace. py is located in the installation package directory of the cluster. Executing the replace. py command requires replacing a coordinator node in the cluster with the cluster installation user dbauser.

explain

- After a successful execution of replace.py, the old distribution (in this example, with Distribution ID 1) will be deleted and a new distribution (with Distribution ID 4) will be generated.

Operating Steps

Step 1: Execute replace. py to replace the installation.

```
$ ./ replace.py --host=172.168.83.15 --freenode=172.168.83.16 --type=data
--dbaUser=gbase --dbaUserPwd=gbasedba --generalDBUser=root --general
DBPwd=****** --overwrite --vcname=vc1
172.168.83.15
Are you sure to replace install these nodes ([Y,y]/[N,n])? y
check ip start .....
check ip end .....

switch cluster mode into READONLY start .....
wait all ddl statement stop .....

all ddl statement stoped
switch cluster mode into READONLY end .....
```

```
delete all fevent log on replace nodes start ......
delete ddl event log on node 172.168.83.15 start
delete ddl event log on node 172.168.83.15 end
delete dml event log on node 172.168.83.15 start
delete dml event log on node 172.168.83.15 end
delete dml storage event log on node 172.168.83.15 start
delete dml storage event log on node 172.168.83.15 end
delete all fevent log on replace nodes end ......

sync dataserver metedata begin ......
copy script to data node begin
copy script to data node end
build data packet begin
build data packet end
copy data packet to target node begin
copy data packet to target node end
extract data packet begin
extract data packet end
sync dataserver metedata end, spend time 38370 ms ......

create distribution begin ......
create distribution end

replace nodes spend time: 75255 ms

synchronize data node metadata success
please rebalance instance then remove old distribution after rebalance comple
te success
Replace gcluster nodes successfully.
```

The completed cluster information is as follows:

```
$ gcadmin showdistribution vc vc1

      Distribution ID: 4 | State: new | Total segment num: 3

   Primary Segment Node IP   Segment ID    Duplicate Segment node IP
============================================================
========
|         172.168.83.11       |       1       |         172.168.83.12        |
------------------------------------------------------------------
|         172.168.83.12       |       2       |         172.168.83.16        |
------------------------------------------------------------------
|         172.168.83.16       |       3       |         172.168.83.11        |
```

```
==========================================================
========

      Distribution ID: 3 | State: old | Total segment num: 3

    Primary Segment Node IP   Segment ID    Duplicate Segment node IP
==========================================================
=========
|          172.168.83.11        |       1       |           172.168.83.12          |
----------------------------------------------------------------
|          172.168.83.12        |       2       |                                  |
----------------------------------------------------------------
|          172.168.83.11        |       3       |                                  |
==========================================================
=========
$ gcadmin showdistribution vc vc1 node
    Distribution ID: 4 | State: new | Total segment num: 3

==========================================================
===
|   nodes    | 172.168.83.11 | 172.168.83.12  |  172.168.83.16 |
------------------------------------------------------------
| primary  |         1           |         2           |          3          |
| segments |                     |                     |                     |
------------------------------------------------------------
|duplicate |         3           |         1           |          2          |
|segments 1|                     |                     |                     |
==========================================================
===


    Distribution ID: 3 | State: old | Total segment num: 3

=====================================================
|   nodes    | 172.168.83.11  |       172.168.83.12       |
-----------------------------------------------------
| primary  |         1           |              2              |
| segments |         3           |                             |
-----------------------------------------------------
|duplicate |                     |              1              |
|segments 1|                     |                             |
=====================================================
```

4.5.3.3.1.6Perform data redistribution

Function Description

Execute the rebalance instance command to redistribute the data to the newly created distribution using freenode.

be careful

- This data redistribution will be based on the actual data redistribution;
- The time required for redistribution needs to be evaluated based on the comprehensive situation of data volume, system CPU, disk, network, etc.

Operating Steps

Step 1: Execute the rebalance instance command to redistribute the data to the newly created distribution (Distribution=4).

```
$ gccli

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Right
s Reserved.

gbase> use vc vc1;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)

gbase> rebalance instance;
Query OK, 3 rows affected (Elapsed: 00:00:01.20)

gbase> select * from gclusterdb.rebalancing_ status;
+------------+---------+------------+----------+---------------------------+-------------------
---------+-----------+------------+----------+---------------+-----------------+
| index_ name | db_ name | table_ name | tmptable | start_ time
          | end_ time                  | status       | percentage | priority |
host               | distribution_ id |
+------------+---------+------------+----------+---------------------------+-------------------
---------+-----------+------------+----------+---------------+-----------------+
| demo.t       | demo       | t              |              | 2020-07-29 18:31:39.332
000 | 2020-07-29 18:31:41.392000 | COMPLETED |            100 |            5
 | 172.168.83.11 |                    4 |
| demo.ttt     | demo       | ttt            |              | 2020-07-29 18:31:39.33600
0 | 2020-07-29 18:31:41.389000 | COMPLETED |            100 |            5 |
 172.168.83.11 |                    4 |
| demo.tt      | demo       | tt             |              | 2020-07-29 18:31:39.3360
00 | 2020-07-29 18:31:41.430000 | COMPLETED |            100 |            5
```

```
| 172.168.83.11 |                    4 |
+------------+---------+------------+----------+--------------------------+------------------
---------+-----------+------------+----------+---------------+----------------+
3 rows in set (Elapsed: 00:00:00.03)

gbase> quit
Bye
```

4.5.3.3.1.7Delete old distribution

Function Description

After ensuring that all data rebalances are completed, the old distribution can be deleted and the replaced nodes can be removed from the virtual cluster.

Operating Steps

Step 1: Delete the old distribution (Distribution ID 3) and remove the replaced node from the virtual cluster.

```
$ gcadmin rmdistribution 3 vc vc1
cluster distribution ID [3]
it will be removed now
please ensure this is ok, input [Y,y] or [N,n]: y
select count(*) from gbase.nodedatamap where data_ distribution_ id=3 result
 is not 0
refreshnodedatamap drop 3 success
gcadmin remove distribution [3] success

The completed cluster information is as follows:
$ gcadmin
CLUSTER STATE:              ACTIVE

=============================================================
=====
|                GBASE COORDINATOR CLUSTER INFORMATION
      |
=============================================================
=====
|    NodeName    |    IpAddress    | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
```

```
| coordinator3 | 172.168.83.13 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
=============================================================
|              GBASE VIRTUAL CLUSTER INFORMATION              |
=============================================================
|     VcName     | DistributionId |          comment          |
-------------------------------------------------------------
|       vc1      |          4     | comment message vc1 |
-------------------------------------------------------------
|       vc2      |          2     | comment message vc2 |
-------------------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node

$ gcadmin showdistribution vc vc1

      Distribution ID: 4 | State: new | Total segment num: 3

      Primary Segment Node IP       Segment ID    Duplicate Segment node
 IP
=============================================================
==============
|            172.168.83.11          |       1       |          172.168.83.12
 |
-------------------------------------------------------------------------
|            172.168.83.12          |       2       |          172.168.83.16
 |
-------------------------------------------------------------------------
|            172.168.83.16          |       3       |          172.168.83.11
 |
=============================================================
==============
[ gbase@gba01   gcinstall]$ gcadmin showdistribution vc vc1 node
      Distribution ID: 4 | State: new | Total segment num: 3

=============================================================
============
|   nodes    |    172.168.83.11   |    172.168.83.12   |    172.168.83.16     |
-------------------------------------------------------------------------
| primary    |          1         |          2         |          3
 |
| segments |                      |                    |
```

```
 |
-------------------------------------------------------------------
|duplicate |          3          |          1          |          2          |

|segments  1|                     |                     |
 |
=============================================================
============
```

4.5.3.3.2 Replacing Pure Data Nodes with New Nodes

Replacement instructions

- Create a new distribution, redistribute data from the old distribution to the newly created distribution after successful node replacement, with the sharding on non replacement nodes unchanged;
- The data on the replaced node must have at least one available backup. During the replacement operation, check if there are available shards. If there are no available shards, an error will be reported when performing node replacement;
- The replacement command is executed on the cluster coordinator node. In the cluster status displayed by gcadmin, the GCLUSTER and GCWARE process state on the execution node must be OPEN;
- After replacing with a new node, the node IP remains unchanged and the distribution rules remain unchanged;
- When replacing a pure data node of a VC, it can access other VCs.

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13

Data node:

vc1：172.168.83.11，172.168.83.12，172.168.83.15

Vc2：172.168.83.13，172.168.83.14

Replace 172.168.83.15 with a new machine.

4.5.3.3.2.1Prepare a new node environment

Prepare the operating system environment for the new machine according to the installation chapter.

explain

- The operating system version of the new node is the same as the current node to be replaced.
- The IP of the new node is the same as the IP of the current node to be replaced.

4.5.3.3.2.2Set node status and clean feventlog

Check the node status, the cluster status should be normal, and the coordinator node status should be normal. After confirming that the replaced node is a pure Data node, set the replaced node status to unavailable.

Operating Steps

Step 1: Check the node status. The cluster status should be normal, the coordinator node status should be normal, and the replaced node should be DATA NODE.

```
$gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL


=====================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                |
=====================================================
|       VcName       | DistributionId |            comment            |
-----------------------------------------------------
|         vc1          |            1             | comment message vc1 |
-----------------------------------------------------
=============================================================================
==========
|                                  VIRTUAL CLUSTER DATA NODE INFORMATION
|
=============================================================================
==========
| NodeName |      IpAddress         | DistributionId |       gnode       |syncserver|DataState |
-----------------------------------------------------------------------------
|   node1     | 172.168.83.11     |            1             |       OPEN        |     OPEN     |       0        |
-----------------------------------------------------------------------------
|   node2     | 172.168.83.12     |            1             |       OPEN        |     OPEN     |       0        |
-----------------------------------------------------------------------------
|   node3     | 172.168.83.15     |            1             |       OPEN        |     OPEN     |       0        |
-----------------------------------------------------------------------------


3 data node
```

Step 2: Set the status of the replaced node to unavailable: Run the gcadmin setnodestate command under the DBA user (specified by the dbauser parameter in the demo. options file) of the operating system to set the status of the node to be replaced to unavailable.

```
$ gcadmin setnodestate 172.168.83.15 unavailable
after set node state into unavailable,can not set the state into normal,
```

```
must run gcadmin replacenodes to replace this node ,after that command node state can return into normal.
you realy want to set node state into unavailable(yes or no)?
yes
get node data state by ddl fevent log start ......
get node data state by ddl fevent log end ......
get node data state by dml fevent log start ......
get node data state by dml fevent log end ......
get node data state by dml storage fevent log start ......
get node data state by dml storage fevent log end ......

check data server node data state by fevent log start ......
check data server node data state by fevent log end ......
set node [172.168.83.15] state to unavailable successful
```

View cluster status:

```
$gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

=========================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                |
=========================================================
|       VcName       | DistributionId |            comment            |
---------------------------------------------------------
|        vc1          |            1             | comment message vc1 |
---------------------------------------------------------
=============================================================================
==========
|                                VIRTUAL CLUSTER DATA NODE INFORMATION
|
=============================================================================
==========
| NodeName |     IpAddress         | DistributionId |       gnode       |syncserver|DataState |
-----------------------------------------------------------------------------------
|   node1     | 172.168.83.11     |            1             |       OPEN        |     OPEN     |       0        |
-----------------------------------------------------------------------------------
|   node2     | 172.168.83.12     |            1             |       OPEN        |     OPEN     |       0        |
```

```
--------------------------------------------------------------------------------
|   node3    | 172.168.83.15    |          1          | UNAVAILABLE |              |              |
--------------------------------------------------------------------------------


3 data node
```

Step 3: Delete the feventlog of the replaced node.

```
$ gcadmin rmfeventlog 172.168.83.15
after rmfeventlog 172.168.83.15, fevent log will be removed, must run gcadmin replacenodes to replace this node.
you realy want to remove node 172.168.83.15 fevent log(yes or no)?
yes
delete ddl event log on node 172.168.83.15 start
delete ddl event log on node 172.168.83.15 end
delete dml event log on node 172.168.83.15 start
delete dml event log on node 172.168.83.15 end
delete dml storage event log on node 172.168.83.15 start
delete dml storage event log on node 172.168.83.15 end
```

4.5.3.3.2.3Create an intermediate distribution

Establish a new distribution, which is used to eliminate replaced nodes while maintaining the sharded distribution of other nodes.

Operating Steps

Step 1: View the distribution information of vc1 where node 172.168.83.15 is located.

```
$ gcadmin showdistribution vc vc1 node
        Distribution ID: 1 | State: new | Total segment num: 3


============================================================
====
|   nodes    |   172.168.83.11   | 172.168.83.12 | 172.168.83.15   |
------------------------------------------------------------
| primary   |          1          |         2          |         3          |
| segments |                        |                    |                    |
------------------------------------------------------------
|duplicate |          3          |         1          |         2          |
|segments 1|                        |                    |                    |
============================================================
====
```

Step 2: Use the gcadmin getdistribution command to save the distribution information of the VC where the node to be replaced is located in the specified file.

From the execution result of step 1, it can be seen that the Distribution ID is 1. Save the distribution information with Distribution ID 1 on vc1 to the file Distribution_ info_ In vc1. xml:

```
$ gcadmin getdistribution 1 distribution_ info_ vc1.xml vc vc1
gcadmin getdistribution 1 distribution_ info_ vc1.xml vc vc1 ...

get segments information
write segments information to file [distribution_info_vc1.xml]

gcadmin getdistribution information successful

$ cat distribution_ info_ vc1.xml
<? xml version='1.0' encoding="utf-8"?>
<distributions>
    <distribution>
        <segments>
            <segment>
                <primarynode ip="172.168.83.11"/>

                <duplicatenodes>
                    <duplicatenode ip="172.168.83.12"/>
                </duplicatenodes>
            </segment>

            <segment>
                <primarynode ip="172.168.83.12"/>

                <duplicatenodes>
                    <duplicatenode ip="172.168.83.15"/>
                </duplicatenodes>
            </segment>

            <segment>
                <primarynode ip="172.168.83.15"/>

                <duplicatenodes>
                    <duplicatenode ip="172.168.83.11"/>
```

```
                    </duplicatenodes>
                </segment>
            </segments>
        </distribution>
</distributions>
```

Step 3: Modify the distribution rule information for the new distribution.

The modification principle is to ensure that the replaced node has no sharding, and the distribution rules of sharding for other nodes remain unchanged,

- If the shard stored by the replaced node is used as the primary shard, modify the backup shard node of the shard to the primary shard node, that is, if the node IP is in the primary node label, replace the IP in the duplicate nodes label of the segment with the IP of the replaced node, and delete the duplicate nodes label.
- If the shards stored by the replaced node are used as backup shards, that is, if the IP of the replaced node is in the duplicate nodes label, the duplicate nodes label will be deleted.

Modified distribution_ info_ The vc1. XML file reference is as follows:

```
$ cat distribution_ info_ vc1.xml
<? xml version='1.0' encoding="utf-8"?>
<distributions>
    <distribution>
        <segments>
            <segment>
                <primarynode ip="172.168.83.11"/>

                <duplicatenodes>
                    <duplicatenode ip="172.168.83.12"/>
                </duplicatenodes>
            </segment>

            <segment>
                <primarynode ip="172.168.83.12"/>
            </segment>

            <segment>
                <primarynode ip="172.168.83.11"/>
            </segment>
        </segments>
    </distribution>
```

```
</distributions>
```

Step 4: Modify the gcChangeInfo required to create a distribution_ VC1. XML file.

```
$ cat gcChangeInfo_ vc1.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
  <cfgFile file="distribution_info_vc1.xml"/>
</servers>
```

Step 5: Execute to create a new distribution (after executing the example command, a distribution with a Distribution ID of 3 will be created).

```
$ gcadmin distribution gcChangeInfo_ vc1.xml vc vc1
gcadmin generate distribution ...

gcadmin generate distribution successful
The completed cluster information is as follows:
$ gcadmin showdistribution vc vc1

        Distribution ID: 3 | State: new | Total segment num: 3

  Primary Segment Node IP    Segment ID     Duplicate Segment node IP
================================================================
=======
|       172.168.83.11         |      1         |         172.168.83.12           |
----------------------------------------------------------------
|       172.168.83.12         |      2         |                                 |
----------------------------------------------------------------
|       172.168.83.11         |      3         |                                 |
================================================================
=======

          Distribution ID: 1 | State: old | Total segment num: 3

   Primary Segment Node IP Segment ID     Duplicate Segment node IP
================================================================
=======
|         172.168.83.11       |      1         |         172.168.83.12           |
----------------------------------------------------------------
|         172.168.83.12       |      2         |         172.168.83.15           |
```

```
-----------------------------------------------------------------
|          172.168.83.15          |       3          |          172.168.83.11              |
=================================================================
=======

$ gcadmin showdistribution vc vc1 node
        Distribution ID: 3 | State: new | Total segment num: 3

=================================================================
======
|   nodes     |   172.168.83.11   |            172.168.83.12                    |
-----------------------------------------------------------------
| primary   |           1              |                  2                            |
| segments |           3              |                                                |
-----------------------------------------------------------------
|duplicate |                          |                  1                            |
|segments 1|                          |                                                |
=================================================================
=====

        Distribution ID: 1 | State: old | Total segment num: 3

=================================================================
======
|   nodes     |   172.168.83.11 | 172.168.83.12   | 172.168.83.15       |
-----------------------------------------------------------------
| primary   |           1            |         2            |         3              |
| segments |                          |                      |                        |
-----------------------------------------------------------------
|duplicate |           3            |         1            |         2              |
|segments 1|                          |                      |                        |
=================================================================
======
```

4.5.3.3.2.4Initialize hashmap and perform data redistribution

Execute the initnodeDatamap command to initialize the hashmap, and then redistribute the data to the latest distribution (Distribution ID: 2) through the rebalance instance command.

explain

- According to the distribution rules, this rebalance operation will not actually perform data movement, so it will be completed quickly;
- Do not delete the old version of nodeDatamap and distribution after this rebalance operation.

Operating Steps

Step 1:

Step 1: Initialize hashmap:

```
$ gccli -uroot
GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights Reserved.

gbase> use vc vc1;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> initnodedatamap;
Query OK, 0 rows affected, 5 warnings (Elapsed: 00:00:01.45)
```

Step 2: Perform data redistribution:

```
gbase> rebalance instance;
Query OK, 3 rows affected (Elapsed: 00:00:05.60)
View Rebalance Status:
gbase> select index_ name,status,percentage,priority,host,distribution_ id from gclusterdb.rebalancing_ status;
+------------+-----------+------------+----------+---------------+-----------------+
| index_ name | status      | percentage | priority | host              | distribution_ id |
+------------+-----------+------------+----------+---------------+-----------------+
| demo.tt      | COMPLETED |          100 |          5 | 172.168.83.11 |                3 |
| demo.t       | COMPLETED |          100 |          5 | 172.168.83.11 |                3 |
| demo.ttt     | COMPLETED |          100 |          5 | 172.168.83.11 |                3 |
+------------+-----------+------------+----------+---------------+-----------------+
3 rows in set (Elapsed: 00:00:00.17)

gbase> quit
Bye
```

4.5.3.3.2.5Execute node replacement command

Function Description

Replace. py is located in the installation package directory of the cluster. Executing the replace. py command requires replacing a coordinator node in the cluster with the cluster installation user dbauser.

explain

- After successful execution of replace.py, the old distribution (in this example, the distribution with Distribution ID 1) will be deleted and a new distribution (with Distribution ID 4) will be generated.

Operating Steps

Step 1: Unplug the machine network cable of the replaced cluster node (formerly 172.168.83.15) and bring the new machine to the line.

Step 2: Execute replace. py to replace the installation.

```
$ ./ replace.py --host=172.168.83.15 --type=data --dbaUser=gbase --dbaUserPwd=gbasedba --generalDBUser=root --generalDBPwd=****** --overwrite --vcname=vc1

172.168.83.15
Are you sure to replace install these nodes ([Y,y]/[N,n])? [Y,y] or [N,n] : y


Starting all gcluster nodes...
check ip start ......
check ip end ......

switch cluster mode into READONLY start ......
wait all ddl statement stop ......

all ddl statement stoped
switch cluster mode into READONLY end ......

delete all fevent log on replace nodes start ......
delete ddl event log on node 172.168.83.15 start
delete ddl event log on node 172.168.83.15 end
delete dml event log on node 172.168.83.15 start
delete dml event log on node 172.168.83.15 end
delete dml storage event log on node 172.168.83.15 start
delete dml storage event log on node 172.168.83.15 end
delete all fevent log on replace nodes end ......

sync dataserver metedata begin ......
copy script to data node begin
copy script to data node end
build data packet begin
build data packet end
copy data packet to target node begin
copy data packet to target node end
```

```
extract data packet begin
extract data packet end
sync dataserver metedata end, spend time 43145 ms .....

create distribution begin .....
restore node state start .....
restore node state end .....
create distribution end

replace nodes spend time: 75924 ms

synchronize data node metadata success
please rebalance instance then remove old distribution after rebalance comple
te success
Replace gcluster nodes successfully.
```

The completed cluster information is as follows:

$ gcadmin showdistribution vc vc1

```
     Distribution ID: 4 | State: new | Total segment num: 3

  Primary Segment Node IP   Segment ID   Duplicate Segment node IP
=============================================================
======
|        172.168.83.11       |       1        |        172.168.83.12        |
-------------------------------------------------------------
|        172.168.83.12       |       2        |        172.168.83.15        |
-------------------------------------------------------------
|        172.168.83.15       |       3        |        172.168.83.11        |
=============================================================
======

     Distribution ID: 3 | State: old | Total segment num: 3
 Primary Segment Node IP   Segment ID   Duplicate Segment node IP
=============================================================
======
|       172.168.83.11        |       1        |          172.168.83.12          |
-------------------------------------------------------------
|       172.168.83.12        |       2        |                                 |
-------------------------------------------------------------
|       172.168.83.11        |       3        |                                 |
=============================================================
======
```

$ gcadmin showdistribution vc vc1 node

```
    Distribution ID: 4 | State: new | Total segment num: 3
```

```
============================================================
===
|   nodes    | 172.168.83.11 | 172.168.83.12   |   172.168.83.15 |
------------------------------------------------------------
| primary   |         1          |         2          |          3          |
| segments |                     |                    |                     |
------------------------------------------------------------
|duplicate |         3          |         1          |          2          |
|segments  1|                    |                    |                     |
============================================================
===


    Distribution  ID:  3  |  State:  old  |  Total  segment  num:  3


=========================================================
|   nodes    | 172.168.83.11   |        172.168.83.12        |
---------------------------------------------------------
| primary   |         1          |               2               |
| segments |         3          |                               |
---------------------------------------------------------
|duplicate |                    |               1               |
|segments  1|                    |                               |
=========================================================
```

4.5.3.3.2.6Perform data redistribution

Function Description

Execute the rebalance instance command to redistribute the data to the newly created distribution using freenode.

be careful

- This data redistribution will be based on the actual data redistribution;
- The time required for redistribution needs to be evaluated based on the comprehensive situation of data volume, system CPU, disk, network, etc.

Operating Steps

Step 1: Execute the rebalance instance command to redistribute the data to the newly created distribution (Distribution=4).

```
$ gccli
```

```
GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Right
s Reserved.

gbase> use vc vc1;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)

gbase> rebalance instance;
Query OK, 3 rows affected (Elapsed: 00:00:01.20)

gbase> select * from gclusterdb.rebalancing_ status;
+------------+---------+------------+----------+---------------------------+-------------------
---------+-----------+------------+----------+---------------+-----------------+
| index_ name | db_ name | table_ name | tmptable | start_ time
          | end_ time                          | status       | percentage | priority |
host                | distribution_ id |
+------------+---------+------------+----------+---------------------------+-------------------
---------+-----------+------------+----------+---------------+-----------------+
| demo.t        | demo       | t               |               | 2020-07-29 18:31:39.332
000 | 2020-07-29 18:31:41.392000 | COMPLETED |              100 |              5
 | 172.168.83.11 |                        4 |
| demo.ttt      | demo       | ttt             |               | 2020-07-29 18:31:39.33600
0 | 2020-07-29 18:31:41.389000 | COMPLETED |              100 |              5 |
 172.168.83.11 |                        4 |
| demo.tt       | demo       | tt              |               | 2020-07-29 18:31:39.3360
00 | 2020-07-29 18:31:41.430000 | COMPLETED |              100 |              5
| 172.168.83.11 |                        4 |
+------------+---------+------------+----------+---------------------------+-------------------
---------+-----------+------------+----------+---------------+-----------------+
3 rows in set (Elapsed: 00:00:00.03)

gbase> quit
Bye
```

4.5.3.3.2.7Delete old distribution

Function Description

After ensuring that all data rebalances are completed, the old distribution can be deleted and the replaced nodes can be removed from the virtual cluster.

Operating Steps

Step 1: Delete the old distribution (Distribution ID 3) and remove the replaced node from the virtual cluster.

```
$ gcadmin rmdistribution 3 vc vc1
cluster distribution ID [3]
it will be removed now
please ensure this is ok, input [Y,y] or [N,n]: y
select count(*) from gbase.nodedatamap where data_ distribution_ id=3 result
 is not 0
refreshnodedatamap drop 3 success
gcadmin remove distribution [3] success
```

The completed cluster information is as follows:

```
$ gcadmin
CLUSTER STATE:              ACTIVE

=============================================================
=====
|                GBASE COORDINATOR CLUSTER INFORMATION
       |
=============================================================
=====
|    NodeName    |    IpAddress    | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
=========================================================
|              GBASE VIRTUAL CLUSTER INFORMATION              |
=========================================================
|      VcName      | DistributionId |          comment          |
---------------------------------------------------------
|       vc1        |          4          | comment message vc1 |
---------------------------------------------------------
|       vc2        |          2          | comment message vc2 |
---------------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node

$ gcadmin showdistribution vc vc1

 Distribution ID: 4 | State: new | Total segment num: 3
```

```
 Primary Segment Node IP     Segment ID     Duplicate Segment node IP
=============================================================
=======
|       172.168.83.11        |        1         |            172.168.83.12        |
-------------------------------------------------------------------
|       172.168.83.12        |        2         |            172.168.83.15        |
-------------------------------------------------------------------
|       172.168.83.15        |        3         |            172.168.83.11        |
=============================================================
=======
$ gcadmin showdistribution vc vc1 node
      Distribution ID: 4 | State: new | Total segment num: 3


=============================================================
======
|   nodes    | 172.168.83.11   |    172.168.83.12   |    172.168.83.15|
-------------------------------------------------------------------
| primary   |         1             |           2             |           3          |
| segments |                       |                       |                      |
-------------------------------------------------------------------
|duplicate |         3             |           1             |           2          |
|segments  1|                       |                       |                      |
=============================================================
======
```

4.5.3.4 **Replacement of pure coordinator nodes**

Replacement instructions

- Check the status of cluster nodes, the cluster status should be normal;
- The replaced management node is inaccessible, and other management nodes can access it normally;
- The replacement command is executed on the coordinator node of the cluster. When gcadmin displays the cluster status, the GCLUSTERD and GCWARE process state on the execution node must be OPEN;
- The presence of FEVENTLOG information on the replaced node does not affect node replacement;
- During the replacement process, except for the unavailability of this node, other Coordinator nodes can still provide external services without affecting customer business;

- If a node is set to the Unavailable state, the node state must be successfully replaced before it can be restored to normal. The setting operation is automatically completed by the node replacement.

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13

Data node:

vc1：172.168.83.11，172.168.83.12

Vc2：172.168.83.14，172.168.83.15

Replace 172.168.83.13 with a new machine.

4.5.3.4.1.1Prepare a new node environment

Prepare the operating system environment for the new machine according to the installation chapter.

explain

- The operating system version of the new node is the same as the current node to be replaced.
- The IP of the new node is the same as the IP of the current node to be replaced.

4.5.3.4.1.2Set node status and clean feventlog

Check the node status, the cluster status should be normal, and the coordinator node status should be normal. After confirming that the replaced node is a pure coordinator node, set the replaced node status to unavailable.

Operating Steps

Step 1: Check the node status. The cluster status should be normal, and the coordinator node status should be normal. Confirm that the replaced node is only the coordinator node.

```
$ gcadmin
CLUSTER STATE:              ACTIVE

=============================================================
======
|                  GBASE COORDINATOR CLUSTER INFORMATION
|
=============================================================
======
|    NodeName    |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
```

```
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
=============================================================
|                    GBASE VIRTUAL CLUSTER INFORMATION                    |
=============================================================
|       VcName        | DistributionId |                comment                |
-----------------------------------------------------------
|         vc1         |            1            | comment message for vc1 |
-----------------------------------------------------------
|         vc2         |            2            | comment message for vc2 |
-----------------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node

$ gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

=============================================================
|                    GBASE VIRTUAL CLUSTER INFORMATION                    |
=============================================================
|     VcName        | DistributionId |                comment                |
-----------------------------------------------------------
|       vc1           |            1            | comment message for vc1 |
-----------------------------------------------------------
=============================================================
=======
|              VIRTUAL CLUSTER DATA NODE INFORMATION
|
=============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-------------------------------------------------------------
| node1    |172.168.83.11|          1            |OPEN |     OPEN     |       0       |
-------------------------------------------------------------
| node2    |172.168.83.12|          1            |OPEN |     OPEN     |       0       |
-------------------------------------------------------------
```

```
2 data node

$ gcadmin showcluster vc vc2
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

=============================================================
|                  GBASE VIRTUAL CLUSTER INFORMATION                    |
=============================================================
|       VcName       | DistributionId |               comment               |
-------------------------------------------------------------
|         vc2          |            2             | comment message for vc2 |
-------------------------------------------------------------
=============================================================
=======
|                        VIRTUAL CLUSTER DATA NODE INFORMATION
|
=============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-----------------------------------------------------------------
| node1    |172.168.83.14|          2            |OPEN |      OPEN     |       0       |
-----------------------------------------------------------------
| node2    |172.168.83.15|          2            |OPEN |      OPEN     |       0       |
-----------------------------------------------------------------

2 data node
```

Step 2: Set the status of the replaced node to unavailable: Run the gcadmin setnodestate command under the DBA user (specified by the dbauser parameter in the demo. options file) of the operating system to set the status of the node to be replaced to unavailable.

```
$ gcadmin setnodestate 172.168.83.13 unavailable
after set node state into unavailable,can not set the state into normal,
must run gcadmin replacenodes to replace this node ,after that command node state
can return into normal.
you realy want to set node state into unavailable(yes or no)?
yes
get node data state by ddl fevent log start ......
get node data state by ddl fevent log end ......
get node data state by dml storage fevent log start ......
get node data state by dml storage fevent log end ......
check coordinator node data state by fevent log start ......
check coordinator node data state by fevent log end ......
set node [172.168.83.13] state to unavailable successful
```

View cluster status:

```
$ gcadmin showcluster
CLUSTER STATE:                ACTIVE


=============================================================
=====
|                  GBASE COORDINATOR CLUSTER INFORMATION
|
=============================================================
=====
|     NodeName    |      IpAddress    |      gcware      | gcluster|DataState |
-------------------------------------------------------------
| coordinator1| 172.168.83.11|       OPEN       |     OPEN    |       0         |
-------------------------------------------------------------
| coordinator2| 172.168.83.12|       OPEN       |     OPEN    |       0         |
-------------------------------------------------------------
| coordinator3| 172.168.83.13| UNAVAILABLE|                 |                 |
-------------------------------------------------------------
=============================================================
|                  GBASE VIRTUAL CLUSTER INFORMATION                    |
=============================================================
|       VcName        | DistributionId |                comment                |
-------------------------------------------------------------
|         vc1          |            1             | comment message for vc1 |
-------------------------------------------------------------
|         vc2          |            2             | comment message for vc2 |
-------------------------------------------------------------


2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node
```

Step 3: Delete the feventlog of the replaced node.

```
$ gcadmin rmfeventlog 172.168.83.13
after rmfeventlog 172.168.83.13, fevent log will be removed, must run gcadmin
replacenodes to replace this node.
you realy want to remove node 172.168.83.13 fevent log(yes or no)?
yes
delete ddl event log on node 172.168.83.13 start
delete ddl event log on node 172.168.83.13 end
delete dml event log on node 172.168.83.13 start
delete dml event log on node 172.168.83.13 end
delete dml storage event log on node 172.168.83.13 start
delete dml storage event log on node 172.168.83.13 end
```

4.5.3.4.1.3Execute node replacement command

Function Description

Replace. py is located in the installation package directory of the cluster. Executing the replace. py command requires replacing a coordinator node in the cluster with the cluster installation user dbauser.

Operating Steps

Step 1: Unplug the machine network cable of the replaced cluster node (formerly 172.168.83.13) and bring the new machine to be replaced online.

Step 2: Execute replace. py to replace the installation.

```
]$ ./ replace.py --host=172.168.83.13 --type=coor --dbaUser=gbase --dbaUs
erPwd=gbasedba --generalDBUser=root --generalDBPwd=******
172.168.83.13
Are you sure to replace install these nodes ([Y,y]/[N,n])? y
Starting all gcluster nodes...
check ip start ......
check ip end ......

switch cluster mode into READONLY start ......
wait all ddl statement stop ......

all ddl statement stoped
switch cluster mode into READONLY end ......

delete all fevent log on replace nodes start ......
delete ddl event log on node 172.168.83.13 start
delete ddl event log on node 172.168.83.13 end
delete dml event log on node 172.168.83.13 start
delete dml event log on node 172.168.83.13 end
delete dml storage event log on node 172.168.83.13 start
delete dml storage event log on node 172.168.83.13 end
delete all fevent log on replace nodes end ......

sync coordinator metedata start ......
build data packet start ......
build data packet end ......

copy data packet start ......
copy data packet end ......

copy plugin start ......
copy plugin end ......
```

```
uncompress data packet start ......
uncompress data packet end ......

clear temporary file start ......
clear temporary file end ......
sync coordinator metedata end ......
sync coordinator metedata end,spend time 20991 ms.....

restore node state start ......
restore node state end ......

replace nodes spend time: 51068 ms

all nodes replace success end
Replace gcluster nodes successfully.
The completed cluster information is as follows:

$ $ gcadmin
CLUSTER STATE:              ACTIVE

=============================================================
======
|                GBASE COORDINATOR CLUSTER INFORMATION
       |
=============================================================
======
|    NodeName    |    IpAddress    | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
=============================================================

|                GBASE VIRTUAL CLUSTER INFORMATION                |

=============================================================

|      VcName      | DistributionId |             comment             |
-------------------------------------------------------------
|       vc1        |          1          | comment message for vc1 |
-------------------------------------------------------------
|       vc2        |          2          | comment message for vc2 |
```

```
------------------------------------------------------------


2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node
```

4.5.3.5 Replacement of composite nodes

Replacing a composite node requires two node replacements, one for the Coordinator service and the other for the Data service.

be careful

- Replacing composite nodes can only be replaced with brand new nodes;
- To replace a composite node, the coordinator service must be replaced first;
- To replace the Data service, it is necessary to confirm that the status of the coordinator node outside of the replaced node is normal.

Cluster environment description:

Coordinator node: 172.168.83.11172.168.83.12172.168.83.13

Data node:

vc1：172.168.83.11，172.168.83.12，172.168.83.15

Vc2：172.168.83.13，172.168.83.14

Replace the composite node 172.168.83.13 of vc2.

4.5.3.5.1.1Prepare a new node environment

Prepare the operating system environment for the new machine according to the installation chapter.

explain

- The operating system version of the new node is the same as the current node to be replaced.
- The IP of the new node is the same as the IP of the current node to be replaced.

4.5.3.5.1.2Set node status and clean feventlog

Check the node status, the cluster status should be normal, the coordinator node status, and then set the replaced node status to unavailable as normal.

Operating Steps

Step 1: Check the node status. The cluster status should be normal and the node status should be normal.

```
$ gcadmin
CLUSTER STATE:              ACTIVE


=============================================================
|                GBASE COORDINATOR CLUSTER INFORMATION                |
=============================================================
|    NodeName    |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |    OPEN    |        0        |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |    OPEN    |        0        |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |    OPEN    |        0        |
-------------------------------------------------------------
=============================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                |
=============================================================
|      VcName       | DistributionId |              comment              |
-------------------------------------------------------------
|        vc1        |            1            | comment message for vc1 |
-------------------------------------------------------------
|        vc2        |            2            | comment message for vc2 |
-------------------------------------------------------------


2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node


$ gcadmin showcluster vc vc1
CLUSTER STATE:              ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL


=============================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                |
=============================================================
```

```
|       VcName       | DistributionId |                comment                |
-------------------------------------------------------------
|        vc1         |            1            | comment message for vc1 |
-------------------------------------------------------------
=============================================================================
========
|                  VIRTUAL CLUSTER DATA NODE INFORMATION
|
=============================================================================
========
| NodeName|      IpAddress    | DistributionId| gnode| syncserver| DataState|
-----------------------------------------------------------------------------
|   node1    | 172.168.83.11|            1            | OPEN |       OPEN     |        0       |
-----------------------------------------------------------------------------
|   node2    | 172.168.83.12|            1            | OPEN |       OPEN     |        0       |
-----------------------------------------------------------------------------
|   node3    | 172.168.83.15|            1            | OPEN |       OPEN     |        0       |
-----------------------------------------------------------------------------

3 data node

$ gcadmin showcluster vc vc2
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

============================================================
|                  GBASE VIRTUAL CLUSTER INFORMATION                    |
============================================================
|       VcName       | DistributionId |                comment                |
-------------------------------------------------------------
|        vc2         |            2            | comment message for vc2 |
-------------------------------------------------------------
=============================================================================
========
|                          VIRTUAL CLUSTER DATA NODE INFORMATION
|
=============================================================================
========
```

```
| NodeName|      IpAddress    | DistributionId| gnode| syncserver| DataState|
-----------------------------------------------------------------------------
|    node1    | 172.168.83.13|            2            | OPEN |       OPEN     |        0       |
-----------------------------------------------------------------------------
|    node2    | 172.168.83.14|            2            | OPEN |       OPEN     |        0       |
-----------------------------------------------------------------------------
2 data node
```

Step 2: Set the status of the replaced node to unavailable: Run the gcadmin setnodestate command under the DBA user (specified by the dbauser parameter in the demo. options file) of the operating system to set the status of the node to be replaced to unavailable.

```
$ gcadmin setnodestate 172.168.83.13 unavailable
after set node state into unavailable,can not set the state into normal,
must run gcadmin replacenodes to replace this node ,after that command node state can
return into normal.
you realy want to set node state into unavailable(yes or no)?
yes
get node data state by ddl fevent log start ......
get node data state by ddl fevent log end ......
get node data state by dml storage fevent log start ......
get node data state by dml storage fevent log end ......
check coordinator node data state by fevent log start ......
check coordinator node data state by fevent log end ......
set node [172.168.83.13] state to unavailable successful
View cluster status:
$ gcadmin showcluster
CLUSTER STATE:                ACTIVE


==============================================================
======
|                        GBASE COORDINATOR CLUSTER INFORMATION
|
==============================================================
======
|     NodeName     |     IpAddress     |     gcware       | gcluster | DataState |
-------------------------------------------------------------------
| coordinator1 | 172.168.83.11 |       OPEN         |     OPEN     |        0        |
-------------------------------------------------------------------
```

```
| coordinator2 | 172.168.83.12 |       OPEN         |     OPEN     |        0        |
-------------------------------------------------------------------
| coordinator3 | 172.168.83.13 | UNAVAILABLE |                |                 |
-------------------------------------------------------------------
============================================================
|                  GBASE VIRTUAL CLUSTER INFORMATION                  |
============================================================
|       VcName       | DistributionId |               comment               |
------------------------------------------------------------
|         vc1          |            1             | comment message for vc1 |
------------------------------------------------------------
|         vc2          |            2             | comment message for vc2 |
------------------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node
```

Step 3: Delete the feventlog of the replaced node.

```
$ gcadmin rmfeventlog 172.168.83.13
after rmfeventlog 172.168.83.13, fevent log will be removed, must run gcadmin replacenodes to replace this node.
you realy want to remove node 172.168.83.13 fevent log(yes or no)?
yes
delete ddl event log on node 172.168.83.13 start
delete ddl event log on node 172.168.83.13 end
delete dml event log on node 172.168.83.13 start
delete dml event log on node 172.168.83.13 end
delete dml storage event log on node 172.168.83.13 start
delete dml storage event log on node 172.168.83.13 end
```

4.5.3.5.1.3Execute node replacement command

Function Description

Replace. py is located in the installation package directory of the cluster. Executing the replace. py command requires replacing a coordinator node in the cluster with the cluster installation user dbauser.

Operating Steps

Step 1: Unplug the machine network cable of the replaced cluster node (formerly 172.168.83.13) and bring the new machine to be replaced online.

Step 2: Execute replace. py to replace the installation.

```
]$ ./ replace.py --host=172.168.83.13 --type=coor --dbaUser=gbase --dbaUserPwd=
gbasedba --generalDBUser=root --generalDBPwd=******
172.168.83.13
Are you sure to replace install these nodes ([Y,y]/[N,n])? y
Starting all gcluster nodes...
check ip start ......
check ip end ......

switch cluster mode into READONLY start ......
wait all ddl statement stop ......

all ddl statement stoped
switch cluster mode into READONLY end ......

delete all fevent log on replace nodes start ......
delete ddl event log on node 172.168.83.13 start
delete ddl event log on node 172.168.83.13 end
delete dml event log on node 172.168.83.13 start
delete dml event log on node 172.168.83.13 end
delete dml storage event log on node 172.168.83.13 start
delete dml storage event log on node 172.168.83.13 end
delete all fevent log on replace nodes end ......

sync coordinator metedata start ......
build data packet start ......
build data packet end ......

copy data packet start ......
copy data packet end ......

copy plugin start ......
copy plugin end ......
uncompress data packet start ......
uncompress data packet end ......

clear temporary file start ......
clear temporary file end ......
sync coordinator metedata end ......
sync coordinator metedata end,spend time 20991 ms.....

restore node state start ......
```

```
restore node state end .....

replace nodes spend time: 51068 ms

all nodes replace success end
Replace gcluster nodes successfully.
The completed cluster information is as follows:

$ gcadmin
CLUSTER STATE:              ACTIVE

=================================================================
|                GBASE COORDINATOR CLUSTER INFORMATION                |

=================================================================
|    NodeName    |    IpAddress    | gcware | gcluster | DataState |
-----------------------------------------------------------------
| coordinator1 | 172.168.83.11 |   OPEN   |    OPEN    |       0       |
-----------------------------------------------------------------
| coordinator2 | 172.168.83.12 |   OPEN   |    OPEN    |       0       |
-----------------------------------------------------------------
| coordinator3 | 172.168.83.13 |   OPEN   |    OPEN    |       0       |
-----------------------------------------------------------------
=============================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                |
=============================================================
|      VcName      | DistributionId |             comment             |
-------------------------------------------------------------
|       vc1        |          1          | comment message for vc1 |
-------------------------------------------------------------
|       vc2        |          2          | comment message for vc2 |
-------------------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node
$ gcadmin showcluster vc vc1
CLUSTER STATE:              ACTIVE
VIRTUAL CLUSTER MODE:   NORMAL

=============================================
|        GBASE VIRTUAL CLUSTER INFORMATION        |
=============================================
|      VcName      | DistributionId |    comment    |
---------------------------------------------
```

```
|        vc1          |           1            | vc1comments |
-----------------------------------------------
================================================================
======
|                VIRTUAL CLUSTER DATA NODE INFORMATION
      |
================================================================
======
| NodeName|   IpAddress   |DistributionId| gnode| syncserver| DataState|
---------------------------------------------------------------
|   node1   |172.168.83.11|          1          | OPEN |      OPEN     |       0      |
---------------------------------------------------------------
|   node2   |172.168.83.12|          1          | OPEN |      OPEN     |       0      |
---------------------------------------------------------------
|   node3   |172.168.83.15|          1          | OPEN |      OPEN     |       0      |
---------------------------------------------------------------

3 data node

$ gcadmin showcluster vc vc2
CLUSTER STATE:              ACTIVE
VIRTUAL CLUSTER MODE:   NORMAL

================================================
|         GBASE VIRTUAL CLUSTER INFORMATION          |
================================================
|      VcName      | DistributionId |     comment     |
------------------------------------------------
|        vc2          |           2            | vc2comments |
------------------------------------------------
================================================================
========
|                VIRTUAL CLUSTER DATA NODE INFORMATION
          |
================================================================
========
|NodeName|   IpAddress   |DistributionId|     gnode     |syncserver|DataState|
-------------------------------------------------------------------
| node1   |172.168.83.13|          2          |UNAVAILABLE|               |             |
-------------------------------------------------------------------
| node2   |172.168.83.14|          2          |     OPEN      |     OPEN     |      0      |
-------------------------------------------------------------------

2 data node
```

4.5.3.5.1.4Check node status and clean feventlog

Check the node status, the cluster status should be normal, the coordinator node status should be normal, and the replaced node's DATA NODE INFORMATION status should be unavailable.

Operating Steps

Step 1: Check the node status. The cluster status should be normal, the coordinator node status should be normal, the replaced node should be DATA NODE, and the status should be unavailable.

```
$ gcadmin showcluster vc vc2
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL


=============================================
|          GBASE VIRTUAL CLUSTER INFORMATION          |
=============================================
|       VcName       | DistributionId |      comment      |
---------------------------------------------
|         vc2          |             2             | vc2comments |
---------------------------------------------
=================================================================
========
|               VIRTUAL CLUSTER DATA NODE INFORMATION
|
=================================================================
========
|NodeName|    IpAddress    |DistributionId|      gnode      |syncserver|DataState|
-------------------------------------------------------------------
| node1    |172.168.83.13|          2           |UNAVAILABLE|                |              |
-------------------------------------------------------------------
| node2    |172.168.83.14|          2           |     OPEN       |     OPEN     |       0       |
-------------------------------------------------------------------

2 data node
```

Step 3: Delete the feventlog of the replaced node.

```
$ gcadmin rmfeventlog 172.168.83.13
after rmfeventlog 172.168.83.13, fevent log will be removed, must run gcadmin
replacenodes to replace this node.
you realy want to remove node 172.168.83.13 fevent log(yes or no)?
yes
delete ddl event log on node 172.168.83.13 start
delete ddl event log on node 172.168.83.13 end
delete dml event log on node 172.168.83.13 start
```

```
delete dml event log on node 172.168.83.13 end
delete dml storage event log on node 172.168.83.13 start
delete dml storage event log on node 172.168.83.13 end
```

4.5.3.5.1.5Create an intermediate distribution

Establish a new distribution, which is used to eliminate replaced nodes while maintaining the sharded distribution of other nodes.

Operating Steps

Step 1: View the distribution information of vc2 where node 172.168.83.13 is located.

```
$ gcadmin showdistribution vc vc2 node
   Distribution ID: 2 | State: new | Total segment num: 2

============================================
|   nodes     |172.168.83.13 |172.168.83.14     |
--------------------------------------------
| primary   |        1             |        2               |
| segments |                       |                        |
--------------------------------------------
|duplicate |        2             |        1               |
|segments 1|                       |                        |
============================================
```

Step 2: Use the gcadmin getdistribution command to save the distribution information of the VC where the node to be replaced is located in the specified file.

From the execution result of step 1, it can be seen that the distribution ID is 2. Save the distribution information with distribution ID 2 on vc2 to the file distribution_ info_ In vc2. xml:

```
$ gcadmin getdistribution 2 distribution_ info_ vc2.xml vc vc2
gcadmin getdistribution 2 distribution_ info_ vc2.xml vc vc2 ...

get segments information
write segments information to file [distribution_info_vc1.xml]

gcadmin getdistribution information successful

$ cat distribution_ info_ vc2.xml
<? xml version='1.0' encoding="utf-8"?>
<distributions>
      <distribution>
            <segments>
                  <segment>
                        <primarynode ip="172.168.83.13"/>
```

```
                    <duplicatenodes>
                        <duplicatenode ip="172.168.83.14"/>
                    </duplicatenodes>
                </segment>

                <segment>
                    <primarynode ip="172.168.83.14"/>

                    <duplicatenodes>
                        <duplicatenode ip="172.168.83.13"/>
                    </duplicatenodes>
                </segment>
            </segments>
        </distribution>
</distributions>
```

Step 3: Modify the distribution rule information for the new distribution.

The modification principle is to ensure that the replaced node has no sharding, and the distribution rules of sharding for other nodes remain unchanged,

- If the shard stored by the replaced node is used as the primary shard, modify the backup shard node of the shard to the primary shard node, that is, if the node IP is in the primary node label, replace the IP in the duplicate nodes label of the segment with the IP of the replaced node, and delete the duplicate nodes label.
- If the shards stored by the replaced node are used as backup shards, that is, if the IP of the replaced node is in the duplicate nodes label, the duplicate nodes label will be deleted.

Modified distribution_ info_ The vc2. XML file reference is as follows:

```
$ cat distribution_ info_ vc2.xml
<? xml version='1.0' encoding="utf-8"?>
<distributions>
    <distribution>
        <segments>
            <segment>
                <primarynode ip="172.168.83.14"/>
            </segment>

            <segment>
                <primarynode ip="172.168.83.14"/>
            </segment>
        </segments>
    </distribution>
```

```
</distributions>
```

Step 4: Modify the gcChangeInfo required to create a distribution_ VC2. XML file.

```
$ cat gcChangeInfo_ vc2.xml
<? xml version="1.0" encoding="utf-8"?>
<servers>
  <cfgFile file="distribution_info_vc2.xml"/>
</servers>
```

Step 5: Execute the creation of a new distribution (distribution with Distribution ID 3).

```
$ gcadmin distribution gcChangeInfo_ vc2.xml vc vc2
gcadmin generate distribution ...

gcadmin generate distribution successful
The completed cluster information is as follows:
$ gcadmin showdistribution vc vc2

          Distribution ID: 3 | State: new | Total segment num: 2

 Primary Segment Node IP        Segment ID     Duplicate Segment node IP
==========================================================
==
|       172.168.83.14           |         1          |                        |
----------------------------------------------------------
|       172.168.83.14           |         2          |                        |
==========================================================
==

          Distribution ID: 2 | State: old | Total segment num: 2

 Primary Segment Node IP        Segment ID     Duplicate Segment node IP
==========================================================
======
|       172.168.83.13           |         1          |       172.168.83.14    |
----------------------------------------------------------------
|       172.168.83.14           |         2          |       172.168.83.13    |
==========================================================
======
$ gcadmin showdistribution vc vc2 node
```

```
            Distribution ID: 3 | State: new | Total segment num: 2


=====================================================
|   nodes     |                  172.168.83.14                  |
-----------------------------------------------------
| primary   |                        1                        |
| segments |                        2                        |
=====================================================



            Distribution ID: 2 | State: old | Total segment num: 2


=========================================================
|   nodes     |   172.168.83.13   |   172.168.83.14          |
---------------------------------------------------------
| primary   |          1          |          2                 |
| segments |                       |                            |
---------------------------------------------------------
|duplicate |          2          |          1                 |
|segments 1|                       |                            |
=========================================================
```

4.5.3.5.1.6Initialize hashmap and perform data redistribution

Execute the initnodeDatamap command to initialize the hashmap, and then redistribute the data to the latest distribution (Distribution ID: 3) through the rebalance instance command.

explain

- According to the distribution rules, this rebalance operation will not actually perform data movement, so it will be completed quickly;
- Do not delete the old version of nodeDatamap and distribution after this rebalance operation.

Operating Steps

Step 1:

Step 1: Initialize hashmap:

```
$ gccli -uroot
GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights Reserved.

gbase> use vc vc2;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> initnodedatamap;
Query OK, 0 rows affected, 5 warnings (Elapsed: 00:00:01.45)
```

Step 2: Perform data redistribution:

```
gbase> rebalance instance;
Query OK, 3 rows affected (Elapsed: 00:00:05.60)
View Rebalance Status:
gbase> select index_ name,status,percentage,priority,host,distribution_ id from gclusterdb.rebalancing_ status;
+------------+-----------+------------+----------+---------------+-----------------+
| index_ name | status      | percentage | priority | host              | distribution_ id |
+------------+-----------+------------+----------+---------------+-----------------+
| demo.tt      | COMPLETED |          100 |          5 | 172.168.83.14 |                3 |
| demo.t       | COMPLETED |          100 |          5 | 172.168.83.14 |                3 |
| demo.ttt     | COMPLETED |          100 |          5 | 172.168.83.14 |                3 |
+------------+-----------+------------+----------+---------------+-----------------+
3 rows in set (Elapsed: 00:00:00.17)

gbase> quit
Bye
```

4.5.3.5.1.7Execute node replacement command

Function Description

Replace. py is located in the installation package directory of the cluster. Executing the replace. py command requires replacing a coordinator node in the cluster with the cluster installation user dbauser.

 **explain**

- After successful execution of replace.py, the old distribution (in this example, the distribution with Distribution ID 2) will be deleted and a new distribution (with Distribution ID 4) will be generated.

Operating Steps

Step 1: Execute replace. py to replace the installation.

```
$ ./ replace.py --host=172.168.83.13 --type=data --dbaUser=gbase --dbaUserPwd=gbasedba --generalDBUser=root --generalDBPwd=****** --overwrite --vcname=vc2
172.168.83.13
Are you sure to replace install these nodes ([Y,y]/[N,n])? y
Starting all gcluster nodes...
check ip start ......
check ip end ......

switch cluster mode into READONLY start ......
wait all ddl statement stop ......

all ddl statement stoped
switch cluster mode into READONLY end ......

delete all fevent log on replace nodes start ......
delete ddl event log on node 172.168.83.13 start
delete ddl event log on node 172.168.83.13 end
delete dml event log on node 172.168.83.13 start
delete dml event log on node 172.168.83.13 end
delete dml storage event log on node 172.168.83.13 start
delete dml storage event log on node 172.168.83.13 end
delete all fevent log on replace nodes end ......

sync dataserver metedata begin ......
copy script to data node begin
copy script to data node end
build data packet begin
build data packet end
copy data packet to target node begin
copy data packet to target node end
extract data packet begin
extract data packet end
sync dataserver metedata end, spend time 20592 ms ......

create distribution begin ......
restore node state start ......
restore node state end ......
create distribution end

replace nodes spend time: 52697 ms

synchronize data node metadata success
please rebalance instance then remove old distribution after rebalance complete succ
```

```
ess
Replace gcluster nodes successfully.

The completed cluster information is as follows:

$ gcadmin showdistribution vc vc2

     Distribution ID: 4 | State: new | Total segment num: 2

  Primary Segment Node IP    Segment ID   Duplicate Segment node IP
=============================================================
===
|       172.168.83.13        |      1      |        172.168.83.14        |
-------------------------------------------------------------
|       172.168.83.14        |      2      |        172.168.83.13        |
=============================================================
===

     Distribution ID: 3 | State: old | Total segment num: 2

  Primary Segment Node IP    Segment ID     Duplicate Segment node IP
=============================================================
======
|       172.168.83.14        |      1      |                             |
-------------------------------------------------------------
|       172.168.83.14        |      2      |                             |
=============================================================
======
$ gcadmin showdistribution vc vc2 node
     Distribution ID: 4 | State: new | Total segment num: 2

=============================================
|   nodes    |  172.168.83.13 | 172.168.83.14 |
---------------------------------------------
| primary    |          1          |        2          |
| segments |                     |                   |
---------------------------------------------
|duplicate |          2          |        1          |
|segments  1|                     |                   |
=============================================


     Distribution ID: 3 | State: old | Total segment num: 2

=================================================
```

```
|   nodes    |                172.168.83.14                |
-------------------------------------------------------
| primary    |                    1                        |
| segments |                    2                        |
=======================================================
```

4.5.3.5.1.8Perform data redistribution

Function Description

Execute the rebalance instance command to redistribute the data to the newly created distribution.

be careful

- This data redistribution will be based on the actual data redistribution;
- The time required for redistribution needs to be evaluated based on the comprehensive situation of data volume, system CPU, disk, network, etc.

Operating Steps

Step 1: Execute the rebalance instance command to redistribute the data to the newly created distribution (Distribution=4).

```
$ gccli

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Right
s Reserved.

gbase> use vc vc2;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)

gbase> rebalance instance;
Query OK, 3 rows affected (Elapsed: 00:00:01.20)

gbase> select * from gclusterdb.rebalancing_ status;
+------------+---------+------------+----------+---------------------------+-------------------
---------+-----------+------------+----------+---------------+-----------------+
| index_ name | db_ name | table_ name | tmptable | start_ time
          | end_ time                         | status      | percentage | priority |
host               | distribution_ id |
+------------+---------+------------+----------+---------------------------+-------------------
---------+-----------+------------+----------+---------------+-----------------+
| demo.t       | demo      | t              |              | 2020-07-29 18:31:39.332
000 | 2020-07-29 18:31:41.392000 | COMPLETED |          100 |           5
 | 172.168.83.14 |                    4 |
```

```
| demo.ttt    | demo       | ttt              |                  | 2020-07-29 18:31:39.33600
0 | 2020-07-29 18:31:41.389000 | COMPLETED |              100 |              5 |
 172.168.83.14 |                          4 |
| demo.tt     | demo       | tt               |                  | 2020-07-29 18:31:39.3360
00 | 2020-07-29 18:31:41.430000 | COMPLETED |              100 |              5
| 172.168.83.14 |                          4 |
+-------------+----------+-------------+-----------+----------------------------+-------------------
----------+------------+-------------+-----------+----------------+-----------------+
3 rows in set (Elapsed: 00:00:00.03)

gbase> quit
Bye
```

4.5.3.5.1.9Delete old distribution

Function Description

After ensuring that all data rebalances are completed, the old distribution can be deleted and the replaced nodes can be removed from the virtual cluster.

Operating Steps

Step 1: Delete the old distribution (Distribution ID 3) and remove the replaced node from the virtual cluster.

```
$ gcadmin rmdistribution 3 vc vc1
cluster distribution ID [3]
it will be removed now
please ensure this is ok, input [Y,y] or [N,n]: y
select count(*) from gbase.nodedatamap where data_ distribution_ id=3 result
 is not 0
refreshnodedatamap drop 3 success
gcadmin remove distribution [3] success

The completed cluster information is as follows:
$ gcadmin
CLUSTER STATE:              ACTIVE

=============================================================
=====
|                GBASE COORDINATOR CLUSTER INFORMATION
      |
=============================================================
=====
|    NodeName    |    IpAddress    | gcware | gcluster | DataState |
```

```
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
=============================================================
|              GBASE VIRTUAL CLUSTER INFORMATION              |
=============================================================
|      VcName      | DistributionId |          comment          |
-------------------------------------------------------
|       vc1        |          1          | comment message vc1 |
-------------------------------------------------------
|       vc2        |          4          | comment message vc2 |
-------------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node

$ gcadmin showdistribution vc vc2

  Distribution ID: 4 | State: new | Total segment num: 2

  Primary Segment Node IP   Segment ID   Duplicate Segment node IP
=============================================================
======
|          172.168.83.13        |      1        |         172.168.83.14         |
-------------------------------------------------------------
|          172.168.83.14        |      2        |         172.168.83.13         |
=============================================================
======
$ gcadmin showdistribution vc vc2 node
     Distribution ID: 4 | State: new | Total segment num: 2

=============================================================
====
|   nodes    | 172.168.83.13 |            172.168.83.14                |
-------------------------------------------------------------
| primary   |         1           |                 2                          |
| segments |                      |                                              |
-------------------------------------------------------------
|duplicate |         2           |                 1                          |
```

```
|segments  1|                    |                                        |
============================================================
====
```

4.5.4 Cluster data redistribution

4.5.4.1 Rebalance command

The rebalance command is responsible for redistributing table data between nodes and supports the following functions:

1) The database provides three levels of data redistribution, namely instance level, database level distribution, and table level. Users can choose appropriate distribution levels based on different scenarios

2) For data movement that requires a large number of tables, distribution priority and concurrent distribution quantity are also provided for finer grained control.

3) The rebalance command returns the results after the task is sent to the background. The database provides a query of the execution process information. You can query the start and end distribution time, distribution progress, distribution priority and other information according to the data table. You can access gclusterdb.rebalancing_ Status to view.

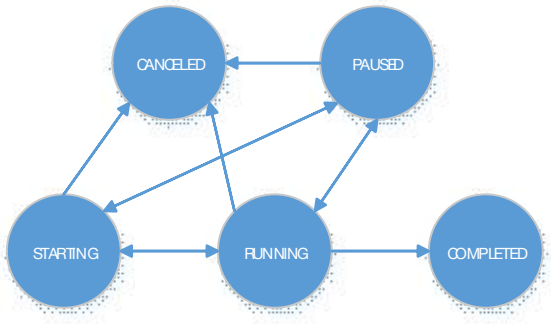
4) Provide cancellation and pause functions to control background tasks.

explain

- After executing the rebalance command using gccli on the coordinator, the rebalance task will be added to gclusterdb. rebalancing_ Status cluster table. The coordinator cluster will be updated from gclusterdb.rebalancing_ Select the gcluster with the highest priority in status_ rebalancing_ concurrent_ Count tables for rebalance.
- There will only be one coordinator node in the coordinator cluster responsible for executing table rebalance in the background, and it is not supported for multiple coordinators to execute rebalance commands in parallel.
- Rebalance task execution status needs to be updated from gclusterdb.rebalancing_ Query in the status table.
- Not recommended for gclusterdb.rebalancing_ Perform ddl/dml operations on the status table.
- Only rebalance for express engine tables is supported

A table has 5 states during rebalance, namely: STARTING, RUNNING, Completed, PAUSED, and CANCELED. These 5 state transitions are shown in the following figure:

Figure -42: 5 state transition diagrams



explain

- When the table is in the STARTING state, the coordinator backend thread starts performing a rebalance operation on the table, and the table state transitions to RUNNING.
- When the table is in the PAUSED state, perform a continue rebalance operation on the table, and the table state transitions to RUNNING.
- When the table is in the RUNNING state, the coordinator backend thread fails to perform a rebalance operation on the table, and the table state transitions to START.
- When the table is in the RUNNING state, the coordinator backend thread completes the rebalance operation of the table, and the table state transitions to Completed.

4.5.4.1.1 rebalance

Function Description

Rebalance is distributed to the latest distribution by default, or it can be distributed to a specified topology using the to distributionid syntax. Before conducting distribution, it is necessary to confirm gclusterdb.rebalancing_ There are no distribution records for this table in the status. If there are, you can use the delete from statement to delete the records, otherwise it will affect the execution of distribution commands.

Grammar format

```
rebalance <rebalance_ options>    [to distribution_id]

rebalance_ options:
    Table [[vc_name.]database_ name.] table_ name
| Database [vc_name.]database_ name
| instance
```

Table -442 Parameter Description

Parameter Name	Description

Parameter Name	Description
table [[vc_name.]database_ name.] table_ name	Convert the specified table from one distribution rule to be distributed according to another distribution rule. If the database_ name.table_ The name has already been distributed on a new distribution (the distribution with a State value of new in the execution result of the gcadmin showdistribution command). The rebalance table operation reported an error and will not report to gclusterdb.rebalancing_ Add Rebalance task in status.
database [vc_name.]database_ name	Batch convert all tables under the specified database from one distribution rule to be distributed according to another distribution rule. If the database_ name.table_ The name has already been distributed on a new distribution (the distribution with a State value of new in the execution result of the gcadmin showdistribution command). The rebalance table operation reports an error and will not report to gclusterdb.rebalancing_ Add Rebalance task in status. Rebalance database returns adding gclusterdb.rebalancing_ The number of rebalance tasks for status.
instance	Batch convert all tables under a specified instance from one distribution rule to be distributed according to another distribution rule. If the database_ name.table_ The name has already been distributed on a new distribution (the distribution with a State value of new in the execution result of the gcadmin showdistribution command). The rebalance table operation reported an error and will not report to gclusterdb.rebalancing_ Add Rebalance task in status. Rebalance instance returns adding gclusterdb.rebalancing_ The number of rebalance tasks for status.
to distribution_ id	If [to distribution_id] is not specified, the rebalance operation will be distributed according to the new distribution rule (i.e. the distribution with the State value new in the gcadmin showdistribution command execution

Parameter Name	Description
	result). If [to distribution_id] is specified, the rebalance operation will follow the specified distribution_ The distribution rule of ID is used for distribution.

Example

Example 1:

```
gbase> rebalance table testdis;
Query OK, 1 row affected
gbase> rebalance table testdis to 1;
Query OK, 1 row affected
```

Example 2:

```
gbase> rebalance database test;
Query OK, 3 rows affected
gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+---------------+----------+------------+
| index_ name       | status      | percentage |
+---------------+----------+------------+
| test.testrep    | RUNNING    | 0                  |
| test.testdis    | STARTING | 0                  |
| test.testrand | STARTING | 0                  |
+---------------+----------+------------+
3 row in set
gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+---------------+-----------+------------+
| index_ name       | status        | percentage |
+---------------+-----------+------------+
| test.testrep    | COMPLETED | 100              |
| test.testdis    | RUNNING      | 10               |
| test.testrand | RUNNING      | 90               |
+---------------+-----------+------------+
3 row in set
```

Example 3:

```
gbase> rebalance instance;
Query OK, 6 rows affected


gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+---------------+----------+------------+
```

```
| index_ name       | status     | percentage |
+---------------+----------+------------+
| test1.t1          | RUNNING    | 0                |
| test.testdis    | STARTING | 0                |
| test.testrand | STARTING | 0                |
| test1.t3          | STARTING | 0                |
| test1.t2          | STARTING | 0                |
| test.testrep    | STARTING | 0                |
+---------------+----------+------------+
6 rows in set
```

4.5.4.1.2 pause rebalance table

Command Description

If the object undergoing a rebalance operation is in a STARTING or RUNNING state, the pause rebalance table command can be used to pause the rebalance operation. If the pause rebalance table command returns 1 affected rows, then the task is successfully paused; If the number of affected rows returned is 0, the task will pause and fail.

be careful

- If the running status progress of the rebalance operation has exceeded 90%, executing the pause command will not work

Grammar format

```
pause rebalance <rebalance_ options>

rebalance_ options:
    Table [[vc_name.]database_ name.] table_ name
| Database [vc_name.]database_ name
| instance
```

Table -443 Parameter Description

Parameter Name	Description
table [[vc_name.]database_ name.] table_ name	Pause the specified rebalance operation in the STARTING or RUNNING state table. The number of affected rows returned by pause rebalance instance is the number of rebalance tasks that have been paused.

Parameter Name	Description
database [vc_name.]database_ name	Pause rebalance database pauses all rebalance operations in the STARTING or RUNNING state tables under the specified database. The number of affected rows returned by the pause rebalance database is the number of rebalance tasks that have been paused.
instance	Pause all rebalance operations in the STARTING or RUNNING state table for the specified instance. The number of affected rows returned by pause rebalance instance is the number of rebalance tasks that have been paused.

Example

Example 1:

```
gbase> select index_ name, status, percentage  from gclusterdb.rebalancing_ status;
+--------------+--------+------------+
| index_ name     | status | percentage |
+--------------+--------+------------+
| test.testdis | RUNNING| 10              |
+--------------+--------+------------+
1 row in set
gbase> pause rebalance table testdis ;
Query OK, 1 row affected
gbase> select index_ name, status, percentage  from gclusterdb.rebalancing_ status;
+--------------+--------+------------+
| index_ name     | status | percentage |
+--------------+--------+------------+
| test.testdis | PAUSED | 10              |
+--------------+--------+------------+
1 row in set
```

Example 2:

```
gbase> select index_ name, status, percentage  from gclusterdb.rebalancing_ status;
+--------------+-----------+------------+
| index_ name      | status       | percentage |
+--------------+-----------+------------+
| test.testrand | COMPLETED | 100             |
| test.testdis   | RUNNING     | 10              |
```

```
| test.testrep    | RUNNING      | 90                |
+---------------+-----------+------------+
3 rows in set

gbase> pause rebalance database test;
Query OK, 1 row affected

gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+---------------+-----------+------------+
| index_ name        | status        | percentage |
+---------------+-----------+------------+
| test.testrand | COMPLETED | 100               |
| test.testdis    | PAUSED        | 30                |
| test.testrep    | COMPLETED | 100               |
+---------------+-----------+------------+
3 rows in set
```

Example 3:

```
gbase> pause rebalance instance;
Query OK, 4 rows affected

gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+---------------+-----------+------------+
| index_ name        | status        | percentage |
+---------------+-----------+------------+
| test1.t1           | PAUSED        | 10                |
| test.testdis    | PAUSED        | 10                |
| test.testrand | COMPLETED | 100               |
| test1.t3           | PAUSED        | 10                |
| test1.t2           | PAUSED        | 10                |
| test.testrep    | COMPLETED | 100               |
+---------------+-----------+------------+
6 rows in set
```

4.5.4.1.3 **continue rebalance table**

Command Description

If the rebalance object is in the PAUSED state, you can use the continue rebalance command to continue rebalancing.

Grammar format

```
continue rebalance    <rebalance_ options>

rebalance_ options:
    Table [[vc_name.]database_ name.] table_ name
| Database [vc_name.]database_ name
| instance
```

Table -444 Parameter Description

Parameter Name	Description
table [[vc_name.]database_ name.] table_ name	Continue rebalance table causes the specified table in the PAUSED state to continue rebalancing.
database [vc_name.]database_ name	Continue Rebalance Database_ Continue rebalancing for all tables in the PAUSED state under name.
instance	Continue rebalance instance causes all tables in the PAUSED state under the specified instance to continue rebalancing.

Example

Example 1:

```
gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+--------------+--------+------------+
| index_ name      | status | percentage |
+--------------+--------+------------+
| test.testdis | PAUSED | 10                 |
+--------------+--------+------------+
1 row in set

gbase> continue rebalance table test.testdis;
Query OK, 1 row affected

gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+--------------+---------+------------+
| index_ name      | status    | percentage |
+--------------+---------+------------+
| test.testdis | RUNNING | 10                 |
+--------------+---------+------------+
1 row in set
```

Example 2:

```
gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+---------------+-----------+------------+
| index_ name       | status       | percentage |
+---------------+-----------+------------+
| test.testrep    | PAUSED       | 0                |
| test.testrand | PAUSED       | 10               |
| test.testdis    | PAUSED       | 10               |
+---------------+-----------+------------+
3 rows in set

gbase> continue rebalance database test;
Query OK, 3 rows affected

gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+---------------+-----------+------------+
| index_ name       | status       | percentage |
+---------------+-----------+------------+
| test.testrep    | STARTING |    0               |
| test.testrand | RUNNING     | 10               |
| test.testdis    | RUNNING     | 20               |
+---------------+-----------+------------+
3 rows in set
```

Example 3:

```
gbase> continue rebalance instance ;
Query OK, 4 rows affected

gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+---------------+-----------+------------+
| index_ name       | status       | percentage |
+---------------+-----------+------------+
| test.testrep    | COMPLETED | 100             |
| test.testdis    | RUNNING     | 10               |
| test.testrand | COMPLETED | 100             |
| test1.t1          | RUNNING     | 10               |
| test1.t3          | RUNNING     | 10               |
| test1.t2          | RUNNING     | 10               |
+---------------+-----------+------------+
6 rows in set
```

4.5.4.1.4 cancel rebalance table

Command Description

If the rebalance object is in the START, RUNNING, or PAUSED state, the cancel rebalance command can be used to terminate the rebalance operation. If the cancel rebalance command returns 1 affected rows, the rebalance operation is terminated successfully; If the number of affected rows is 0, the termination operation fails.

be careful

- If the running status progress of the rebalance operation has exceeded 90%, executing the cancel command will not work

Grammar format

```
cancel rebalance   <rebalance_ options>

rebalance_ options:
    Table [[vc_name.]database_ name.] table_ name
| Database [vc_name.]database_ name
| instance
```

Table 445 Parameter Description

Parameter Name	Description
table [[vc_name.]database_ name.] table_ name	If the specified table is in the START, RUNNING, or PAUSED state, the cancel rebalance command can be used to terminate the rebalance operation. If the cancel rebalance command returns 1 affected rows, the rebalance operation is terminated successfully; If the number of affected rows is 0, the termination operation fails.
database [vc_name.]database_ name	Cancel rebalance database Terminates all rebalance operations in the STARTING, RUNNING, and PAUSED state tables under the specified database. The number of affected rows returned by the cancel rebalance database command is the number of successfully terminated rebalance tasks.

Parameter Name	Description
instance	Cancel rebalance instance Terminates all rebalance operations in the START, RUNNING, and PAUSED state tables under the current instance. The number of affected rows returned by the cancel rebalance instance command is the number of successfully terminated rebalance tasks.

Example

Example 1:

```
gbase> select index_ name, status, percentage  from gclusterdb.rebalancing_ status;
+--------------+--------+------------+
| index_ name     | status | percentage |
+--------------+--------+------------+
| test.testdis | PAUSED | 10              |
+--------------+--------+------------+
1 row in set

gbase> cancel rebalance table testdis;
Query OK, 1 row affected

gbase> select index_ name, status, percentage  from gclusterdb.rebalancing_ status;
+--------------+----------+------------+
| index_ name     | status     | percentage |
+--------------+----------+------------+
| test.testdis | CANCELED | 0               |
+--------------+----------+------------+
1 row in set
```

Example 2:

```
gbase> select index_ name, status, percentage   from gclusterdb.rebalancing_ status;
+---------------+---------+------------+
| index_ name       | status   | percentage |
+---------------+---------+------------+
| test.testdis    | RUNNING | 10              |
| test.testrep    | RUNNING | 90              |
| test.testrand | STARTING| 0               |
+---------------+---------+------------+
3 rows in set
```

```
gbase> cancel rebalance database test ;
Query OK, 3 row affected

gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+---------------+-----------+------------+
| index_ name       | status       | percentage |
+---------------+-----------+------------+
| test.testdis    | CANCELED    | 0                |
| test.testrep    | COMPLETED | 100              |
| test.testrand | CANCELED    | 0                |
+---------------+-----------+------------+
3 rows in set
```

Example 3:

```
gbase> cancel rebalance instance ;
Query OK, 4 rows affected

gbase> select index_ name, status, percentage    from gclusterdb.rebalancing_
status;
+---------------+-----------+------------+
| index_ name       | status       | percentage |
+---------------+-----------+------------+
| test.testrep    | COMPLETED | 100              |
| test.testdis    | CANCELED    | 0                |
| test.testrand | COMPLETED | 100              |
| test1.t1          | CANCELED    | 0                  |
| test1.t3          | CANCELED    | 0                  |
| test1.t2          | CANCELED    | 0                  |
+---------------+-----------+------------+
6 rows in set
```

4.5.4.2 Adjusting Rebalance Task Priorities

Function Description

When performing a rebalance operation in bulk, you can modify gclusterdb.rebalancing_ The status table adjusts the priority of individual rebalance tasks. Do the task with the lowest priority value first.

Grammar format

```
update gclusterdb.rebalancing_ status set priority = <priority_ value> where
index_ name='database_ name.table_ name';
```

Table 446 Parameter Description

Parameter Name	Description
priority_ value	Priority_ The task with the smallest value specified by value is done first, and the default value is 5.
database_ name.table_ name	Specify the name of the table to be prioritized by selecting the index_ name, status, priority from gclusterdb.rebalancing_ The status command is used to view the index in the query results_ The value corresponding to the name column.

Operating Steps

1）Set gcluster_ rebalancing_ concurrent_ The count value is 0.

```
gbase> set global gcluster_ rebalancing_ concurrent_ count = 0;
Query OK, 0 rows affected
```

2）Execute the Rebalance database command to increase the Rebalance task. At this point, all rebalance tasks will not start.

```
gbase> rebalance database test;
Query OK, 3 rows affected
```

3）Adjust the priority of the Rebalance task.

```
gbase> select index_ name, status, priority    from
 gclusterdb.rebalancing_ status;
+------------+----------+----------+
| index_ name | status     | priority |
+------------+----------+----------+
| test.t3       | STARTING |              5 |
| test.t1       | STARTING |              5 |
| test.t2       | STARTING |              5 |
+------------+----------+----------+
3 rows in set
gbase> update gclusterdb.rebalancing_ status set priority = 6 where index_
name='test.t3';
Query OK, 1 row affected
Rows matched: 1    Changed: 1    Warnings: 0

gbase> update gclusterdb.rebalancing_ status set priority = 4 where index_
name='test.t2';
Query OK, 1 row affected
```

```
Rows matched: 1  Changed: 1  Warnings: 0
gbase> select index_ name, status, priority  from
  gclusterdb.rebalancing_ status;
+------------+----------+----------+
| index_ name | status   | priority |
+------------+----------+----------+
| test.t3    | STARTING |        6 |
| test.t1    | STARTING |        5 |
| test.t2    | STARTING |        4 |
+------------+----------+----------+
3 rows in set
gbase> set global gcluster_ rebalancing_ concurrent_ count = 1;
Query OK, 0 rows affected
gbase> select index_ name, status, priority  from  gclusterdb.rebalancing_
status;
+------------+----------+----------+
| index_ name | status   | priority |
+------------+----------+----------+
| test.t3    | STARTING |        6 |
| test.t1    | STARTING |        5 |
| test.t2    | RUNNING  |        4 |
+------------+----------+----------+
3 rows in set
```

4）After adjusting, set gcluster_ rebalancing_ concurrent_ Count is the required number of concurrency

```
gbase> set global gcluster_ rebalancing_ concurrent_ count = 1;
Query OK, 0 rows affected
gbase> select index_ name, status, priority  from  gclusterdb.rebalancing_
status;
+------------+----------+----------+
| index_ name | status   | priority |
+------------+----------+----------+
| test.t3    | STARTING |        6 |
| test.t1    | STARTING |        5 |
| test.t2    | RUNNING  |        4 |
+------------+----------+----------+
3 rows in set
```

4.5.4.3 Rebalance Command Related Parameters

```
gcluster_ rebalancing_ concurrent_ count
```

Meaning: The number of tables that allow concurrent rebalance execution. Set to 0 in the session to indicate that rebalancing is not allowed

Parameter Setting Level	Maximum value	minimum value	Default value
GLOBAL	nothing	0	five

gcluster_ rebalancing_ random_ table_ quick_ mode

Meaning: Use fast mode when performing a rebalance operation on a randomly distributed table.

Parameter Setting Level	Maximum value	minimum value	Default value
GLOBAL	one	0	one

gcluster_ rebalancing_ step

Meaning: Specify the number of redistributed data entries for each batch during the rebalance operation. When the value is 0, the rebalance operation is not batched.

Parameter Setting Level	Maximum value	minimum value	Default value
GLOBAL	nothing	0	one hundred million

gcluster_ rebalancing_ The step parameter value is actually the number of data rows redistributed from each partition and batch of the original table to the intermediate table. The larger the parameter value, the faster the speed of redistributing data from the original table to the intermediate table, and the longer the waiting time during the rebalance process.

If there is little need to pause tasks during the rebalance process, gcluster can be set_ rebalancing_ Step is a larger value. If the task needs to be paused multiple times during the rebalance process, gcluster can be set_ rebalancing_ Step is a smaller value.

gcluster_ rebalancing_ Step estimation method: The number of rows in a single partition of the original table/the expected number of batches.

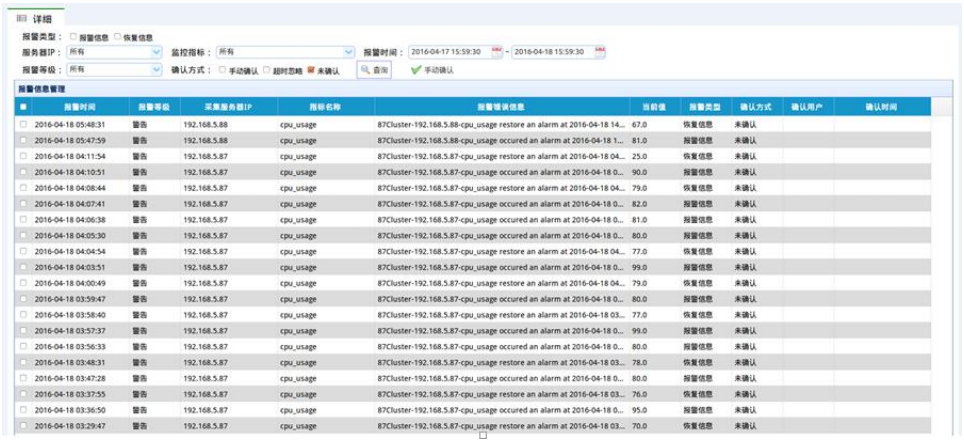
4.6 alarm management

Alarm management is one of the functions of the unified data platform monitoring and operation system, including page alarms, email alarms, and SNMP Trap alarms. Please refer to Chapter 4.1.3- User Manual for Unified Data Platform Monitoring and Operation and Maintenance System for details.

4.6.1 Page alarm

Users can view cluster alarm information through the interface of the unified data platform monitoring and operation maintenance system. As shown in the following figure:

Figure -43 Cluster Alarm Information Management



The query conditions in Figure 4-3 are described as follows:

- Alarm Type: Display multiple selection boxes for alarm types, including alarm information and recovery information. When none is selected by default, checking all indicates that all alarm types are selected.
- Alarm Level: A drop-down box displaying the alarm level, including: Critical, Minor, Warning, Reminder. Default display of alarm information for all alarm levels.
- Server IP: supports multiple selection menus for input retrieval. The first item is "All servers", which is the default value. The others are all server IPs of the cluster, arranged in ascending order.
- Monitoring indicators: drop-down boxes that support multiple selections and fuzzy queries. The indicator item is the indicator item for the selected monitoring strategy of the cluster in platform management.
- Alarm time: supports calendar control selection. The default is the last 24 hours of the current operation. The query time includes the start and end times.
- Confirmation Method: Display a selection box for the alarm information confirmation method, including: manual confirmation, timeout ignore, and unacknowledged. The default is unconfirmed and supports multiple selections. If none is selected, it means all confirmation methods are selected.
- Manual confirmation: After selecting the alarm record, click on manual confirmation. The confirmation method of alarm records can be modified to manual confirmation. Supports multiple selections.

The query results in the above figure are described as follows:

After entering the conditions, click the "Query" button to query the relevant information

and display it. The table defaults to displaying the unconfirmed items first, and then sorting them in descending order of alarm time. After clicking on the table header, you can arrange it in ascending or descending order according to the current column.

The table content is as follows:

- Alarm time: The time when the alarm occurred.
- Alarm level: including serious, minor, warning, and reminder.
- Collection server IP: The IP of the server that generates alarm information.
- Indicator Name: The name of the indicator where the alarm occurred.
- Alarm error message: The specific content of the alarm message. When the mouse moves over the content, a prompt box will pop up, displaying all alarm information.
- Current value: The indicator value of the server when an alarm occurs.
- Alarm type: including alarm information and recovery information. The recovery information refers to the unified monitoring and push of recovery information when the server recovers from an alarm state to a normal state.
- Confirmation methods: including manual confirmation, timeout ignore, and unconfirmed. The initial state of the server when an alarm occurs is unacknowledged; Administrators can confirm alarm information through unified monitoring, and the alarm information status can be changed to manual confirmation; If the current time minus the value of the alarm occurrence time exceeds the timeout ignore time set in the monitoring policy, the system defaults to changing the status of the alarm information to timeout ignore.
- Confirm User: The confirmed user who manually confirms the status is the current user under unified monitoring of operations; The confirmation user for timeout ignoring status is a system administrator.
- Confirmation time: The confirmation time for manually confirming the status is the time when the user manually confirms it; The confirmation time of the timeout ignore status is the time when the system is scheduled to operate.

4.6.2 Email alarm

When adding users, you can set an email address to receive unified monitoring of alarm information emails.

Figure -44 User Information List

4.6.3 SNMP Trap alarm

Unified monitoring provides an SNMP Trap push interface. When the platform discovers an abnormality in a cluster node, Unified monitoring actively pushes the abnormal information to third-party applications through SNMP Trap, so that third-party applications can achieve unified monitoring of the cluster status. This function is not enabled by default and requires modifying the configuration file conf/snmp in the collection center_ udp_ config.properties 。 After modifying the configuration, it is necessary to restart the collection center.

When a node experiences an exception or abnormal recovery, the following PDU content will be pushed:

- 1.3.6.1.4.1.39649.1.9999.1: Node IP where the alarm occurred.
- 1.3.6.1.4.1.39649.1.9999.2: Alarm event code, defined internally for unified monitoring. If it is GBase-09, it indicates that the CPU usage rate has exceeded the threshold and an alarm has occurred.
- 1.3.6.1.4.1.39649.1.999.3: Alarm event description, which describes which monitoring item of which node has triggered an alarm or restored the alarm.
- 1.3.6.1.4.1.39649.1.9999.4: Alarm event name. If it is a CPU_ Usage indicates that an alarm event has occurred regarding CPU usage.
- 1.3.6.1.4.1.39649.1.9999.5: Alarm level, unified monitoring currently supports four alarm levels, including 1: severe; 2: Secondary; 3: Warning; 4: Reminder.
- 1.3.6.1.4.1.39649.1.9999.6: Alarm occurrence time: default time format: yyyy mm dd hh: mm: ss.
- 1.3.6.1.4.1.39649.1.9999.7: Alarm category: 0: Recovery alarm, indicating that the current alarm event is a recovery alarm; 1: Fault alarm: indicates that the current alarm event is a fault alarm.
- 1.3.6.1.4.1.39649.1.9999.8: Event type, 0: No recovery event, indicating that the event does not push recovery information after fault recovery; 1: There is a recovery event, indicating that it will push a recovery event after the fault is recovered.
- 1.3.6.1.4.1.39649.1.999.9: Event Chinese name, such as CPU_ The Chinese name

corresponding to the usage monitoring item is CPU usage.

- 1.3.6.1.4.1.39649.1.999.10: Name of the object where the alarm event occurred. If it is a CPU, it indicates that an alarm has occurred on the CPU.
- 1.3.6.1.4.1.39649.1.9999.11: Alarm threshold, which is the alarm threshold of the monitoring item set in the monitoring website. Like CPU_ The default threshold for usage is 80%, indicating an alarm when the CPU usage exceeds 80%.
- 1.3.6.1.4.1.39649.1.999.12: The current value of the alarm, which is the current value of the alarm monitoring item. Like CPU_ The current cycle collection value of usage is 90%.
- 1.3.6.1.4.1.39649.1.9999.13: The unit of the current value of the alarm item, which is the unit of the monitoring item set in the monitoring website. The unit of CPU usage is%.
- 1.3.6.1.4.1.39649.1.9999.14: Name of the cluster to which the alarm node belongs.

The default format for pushing information is GBK encoding. If you need to modify the encoding format, please go to conf/snmp in the collection center_ udp_ Modify the outputEncoding property in the config.properties file.

4.7 Audit management

4.7.1 Overview of audit logs

Audit logs are used to record user database operations and audit their behavior, mainly for security management. Audit logs will take longer to execute than long _ query_ Recording the time value in SQL facilitates users to analyze, optimize, and rewrite these inefficient SQL statements, thereby improving the execution effi ciency of SQL statements.

4.7.2 Audit Log Parameter Configuration

Operation scenario

Guide administrators to turn on or off audit logs.

Operation method

Execute the following command to open the audit log. It can be controlled through configuration files or global level.

```
SET GLOBAL audit_ log = 1;
```

Execute the following command to set the audit log to be stored in the system table.

```
SET GLOBAL log_ output = 'table';
```

Execute the following command to close the audit log. The default is off.

```
SET GLOBAL audit_ log = 0;
```

4.7.3 Set audit strategy

Operation scenario

Audit policies are used to control the recording of audit logs, which can be set to only record certain specified operations or fixed user actions. This section will guide administrators in creating, modifying, and deleting audit policies.

Operation method

Create audit strategy

The syntax rules for creating audit policies are as follows:

```
CREATE AUDIT POLICY <audit_ policy_ name> [(<audit_policy_item> =
<value>[,<audit_policy_item> = <value>])];
```

Table -447 Parameter Description

Parameter Name	Description
audit_ policy_ name	The name of the audit policy is case insensitive and stored in lowercase. It can include uppercase and lowercase characters, numbers, and underscores, and does not contain special symbols. The first character of the audit name must be an English character.
audit_ policy_ item	For audit strategy projects, audit strategy project names are not case sensitive.

Table -448 Audit Strategy Project Description

entry name	Value&Meaning
Enable	Y: Enable, default value
	N: Disabled
Hosts	": unlimited, default value
	<host>: Strictly match hosts, supporting a list of hosts separated by spaces. Hosts can use "%" and "_" as wildcards
User	": unlimited, default value
	<user>: Strictly match users, case sensitive
Db	": unlimited, default value
	<db>: Strictly match db
Obj_ type	": unlimited, default value
	Table (VIEW): Object is a table (view)
	PROCESS: Object is a stored procedure
	Function: Object is a function
Object	": unlimited, default value
	<object>: Match Obj_ Object specified by type
Sql_ commands	": unlimited, default value
	INSERT, DELETE, UPDATE, LOAD, CREATE_ USER, CREATE_ DB, CREATE_ TABLE, CREATE_ VIEW, CREATE_ INDEX, C

	REATE_ PROCEDURE, CREATE_ FUNCTION, RENAME_ USER, ALTER_ DB, ALTER_ TABLE, ALTER_ PROCEDURE, ALTER_ FUNCTION, ALTER_ EVENT, DROP_ USER, DROP_ DB, DROP_ TABLE, DROP_ VIEW, DROP_ INDEX, DROP_ PROCEDURE, DROP_ FUNCTION, DROP_ EVENT, TRUNCATE, GRANT, REVOKE, SELECT, OTHERS: One or more types, with multiple types connected by commas', 'and no spaces added
Long_ query _ time	<*secs*>: Minimum query seconds, with up to 6 decimal places, accurate to microseconds, default value 0, value range from 0 to 31536 00s
Status	": No restriction, default to no restriction.
	SUCCESS: Execution successful
	FAILED: Execution failed

Modify audit strategy

The syntax rules for modifying audit policies are as follows:

```
ALTER AUDIT POLICY <audit_ policy_ name> SET [(]<audit_policy_item> =
<value>[,<audit_policy_item> = <value>][)];
```

Table 449 Parameter Description

Parameter Name	Description
audit_ policy_ name	The name of the audit policy, case insensitive
audit_ policy_ item	For the audit strategy project, the value content is the same as the description in creating the audit strategy.

Delete Audit Policy

The syntax rules for deleting audit policies are as follows:

```
DROP AUDIT POLICY <audit_ policy_ name>;
```

4.7.4 Store audit logs

The information of the audit log is stored in the system table gbase.audit_ In the log or log gclusterd-audit.log, it depends on the global level variable log_ Configuration of output; If the gbased-audit.log log is not configured, it will be stored in $GCLUSTER by default_ Under the BASE/log/gcluster/directory.

Use constraints

Constraints for using GBase 8a MPP Cluster audit logs:

- Audit logs are used to record all SQL operations. For statistical operations that include the number of rows in the result set, such as SELECT, DELETE, Insert,

UPDATE, MERGE, and ALT;

- Clear audit_ When logging, TRUNCATE SELF audit needs to be used_ Log statement.
- View audit under gclusterdb_ log_ Express can see the logs that have occurred on all nodes.

4.7.5 Audit log high availability

GBase 8a MPP Cluster has a high availability mechanism for audit logs. To achieve high availability of audit logs, during cluster installation or upgrade, the EXPRESS engine random distribution table audit is automatically created under the gclusterdb library_ log_ Express and automatically create a scheduled export event to export the audit in the gbase library_ Regular export of log table content to audit in gclusterdb library_ log_ Express table.

explain

- In the multi VC version, the built-in automatic export event function is invalid, and users need to manually delete and recreate the event.

```
CREATE EVENT "import_audit_log"
        ON SCHEDULE EVERY 60 MINUTE
        STARTS '2017-12-01 00:00:00'
        ON COMPLETION NOT PRESERVE
        ENABLE
        LOCAL
DO
begin
        declare errno int;
        declare msg text;
        declare exit handler for sqlexception
        begin
                get diagnostics condition 1 errno = gbase_ errno, msg = message_ text;
                create table if not exists import_ audit_ log_ errors(
                        err_ time datetime,
                        hostname varchar(64),
                        err_ no int,
                        msg_ txt varchar(1024)
                ) CHARSET=utf8mb4;
                insert into import_ audit_ log_ errors values (now(), @@hostname,
errno, substr(msg, 0, 1024));
        end;
        create table if not exists audit_ log_ express (
```

```
            hostname varchar(64),
            thread_ id int,
            taskid bigint,
            start_ time datetime,
            uid bigint, user varchar(16),
            host_ ip varchar(32),
            query_ time time, rows bigint,
            table_ list varchar(4096),
            sql_ text varchar(8191),
            sql_ type varchar(16),
            sql_ command varchar(32),
            operators varchar(256),
            status varchar(16),
            conn_ type varchar(16)
        ) CHARSET=utf8mb4;
        set self sql_ mode = '';
        create self table gbase.audit_ log_ bak2 like gbase.audit_ log;
        set self sql_ mode = default;
        rename self table gbase.audit_ log to gbase.audit_ log_ bak1, gbase.audit_
log_ bak2 to gbase.audit_ log;
        set _ gbase_ query_ path = on;
        insert into audit_ log_ express select
            @@hostname as hostname,
            thread_ id,
            taskid,
            start_ time,
            uid,
            user,
            host_ ip,
            query_ time,
            rows,
            substr(table_list, 0, 4096),
            substr(sql_text, 0, 8191),
            sql_ type,
            sql_ command,
            operators,
            status,
            conn_ type
        from gbase.audit_ log_ bak1;
        drop self table gbase.audit_ log_ bak1;
end
```

4.7.6 Usage examples

Example

Use system tables to view audit logs.

```
$ gccli -uroot -p
Enter password:
GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.    All Rights
Reserved.
gbase> SET long_ query_ time = 0;
Query OK, 0 rows affected
gbase> SET GLOBAL audit_ log = 1;
Query OK, 0 rows affected
gbase>CREATE AUDIT POLICY audit_ policy_ 1 ( Enable = 'Y' );
Query OK, 0 rows affected
gbase> SET GLOBAL log_ output = 'table';
Query OK, 0 rows affected
gbase> CREATE USER u_ sj identified by 'u_ sj';
Query OK, 0 rows affected
gbase> GRANT ALL ON *.* TO u_ sj;
Query OK, 0 rows affected
gbase> CREATE DATABASE testSJ;
Query OK, 1 row affected
gbase> USE testSJ;
Query OK, 0 rows affected
gbase> CREATE TABLE t1(i int);
Query OK, 0 rows affected
gbase> INSERT INTO t1 VALUES (1),(2);
Query OK, 2 rows affected
gbase> SELECT start_ time,user_ host,query_ time,rows,LEFT(sql_text, 30),
conn_ type FROM gbase.audit_ log;
+---------------------+----------------------------------+
| start_ time         | user_ host                       |
+---------------------+----------------------------------+
| 2021-06-16 10:40:24 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:24 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:24 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:24 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:24 | root[root] @ localhost []            |
| 2021-06-16 10:40:33 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:33 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:33 | root[root] @    [192.168.146.24]     |
```

```
| 2021-06-16 10:40:32 | root[root] @ localhost []              |
| 2021-06-16 10:40:44 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:44 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:44 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:44 | root[root] @ localhost []              |
| 2021-06-16 10:40:48 | gbase[gbase] @    [192.168.146.24] |
| 2021-06-16 10:40:49 | gbase[gbase] @    [192.168.146.24] |
| 2021-06-16 10:40:49 | gbase[gbase] @    [192.168.146.24] |
| 2021-06-16 10:40:57 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:57 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:57 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:57 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:40:57 | root[root] @ localhost []              |
| 2021-06-16 10:41:05 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:41:05 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:41:05 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:41:05 | root[root] @ localhost []              |
| 2021-06-16 10:41:17 | gbase[gbase] @    [192.168.146.24] |
| 2021-06-16 10:41:17 | gbase[gbase] @    [192.168.146.24] |
| 2021-06-16 10:41:17 | gbase[gbase] @    [192.168.146.24] |
| 2021-06-16 10:41:17 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:41:17 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:41:17 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:41:17 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:41:17 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:41:17 | root[root] @ localhost []              |
| 2021-06-16 10:41:29 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:41:29 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:41:29 | root[root] @    [192.168.146.24]     |
| 2021-06-16 10:41:29 | root[root] @ localhost []              |
| 2021-06-16 10:41:43 | root[root] @ localhost []              |
+---------------------+--------------------------------+


+-----------------+------+-------------------------------+-----------+
| query_ time           | rows | LEFT(sql_text, 30)                    | conn_ type |
+-----------------+------+-------------------------------+-----------+
| 00:00:00.000012 |       0 | Ping                                    | CAPI          |
| 00:00:00.000115 |       0 | SET NAMES utf8mb4                       | CAPI          |
| 00:00:00.000164 |       0 | SET SELF SCN = 0                        | CAPI          |
| 00:00:00.000094 |       0 | SET    SELF GLOBAL log_ output =    | CAPI          |
| 00:00:00.239863 |       0 | SET GLOBAL log_ output = 'table | ODBC          |
| 00:00:00.000009 |       0 | Ping                                    | CAPI          |
| 00:00:00.000143 |       0 | SET SELF SCN = 0                        | CAPI          |
```

```
| 00:00:01.149579 |       0 | CREATE GCLUSTER_ LOCAL USER "u_ | CAPI     |
| 00:00:04.376179 |       0 | CREATE USER u_ sj identified by | ODBC           |
| 00:00:00.000019 |       0 | Ping                                          | CAPI           |
| 00:00:00.000207 |       0 | SET SELF SCN = 0                              | CAPI           |
| 00:00:00.057846 |       0 | GRANT /*+vcid*/ GCLUSTER_ LOCAL | CAPI       |
| 00:00:00.266514 |       0 | GRANT ALL ON *.* TO u_ sj             | ODBC           |
| 00:00:00.001372 |       0 | Connect                                       | CAPI           |
| 00:00:00.000005 |       0 | Ping                                          | CAPI           |
| 00:00:00.000002 |       0 | Quit                                          | CAPI           |
| 00:00:00.000008 |       0 | Ping                                          | CAPI           |
| 00:00:00.000090 |       0 | SET NAMES utf8                                | CAPI           |
| 00:00:00.000147 |       0 | SET SELF SCN = 9259                           | CAPI           |
| 00:00:00.261615 |       0 | CREATE GCLUSTER_ LOCAL DATABASE | CAPI |
| 00:00:00.965256 |       0 | CREATE DATABASE testSJ                | ODBC       |
| 00:00:00.000018 |       0 | Ping                                          | CAPI           |
| 00:00:00.000042 |       0 | Init VC Name                                  | CAPI           |
| 00:00:00.000058 |       0 | Init DB                                       | CAPI           |
| 00:00:00.014822 |       0 | USE testSJ                                    | ODBC           |
| 00:00:00.004341 |       0 | Connect                                       | CAPI           |
| 00:00:00.032791 |       0 | select table_ id from informati | CAPI           |
| 00:00:00.000003 |       0 | Quit                                          | CAPI           |
| 00:00:00.000038 |       0 | Ping                                          | CAPI           |
| 00:00:00.000046 |       0 | Init VC Name                                  | CAPI           |
| 00:00:00.000070 |       0 | Init DB                                       | CAPI           |
| 00:00:00.000547 |       0 | SET SELF SCN = 9260                           | CAPI           |
| 00:00:00.781898 |       0 | CREATE    GCLUSTER_ LOCAL    TABLE    | CAPI
|
| 00:00:02.731924 |       0 | CREATE TABLE t1(i int)                | ODBC           |
| 00:00:00.000132 |       0 | commit                                        | CAPI           |
| 00:00:00.000614 |       0 | set autocommit=1                              | CAPI           |
| 00:00:00.000175 |       0 | set autocommit=1                              | CAPI           |
| 00:00:00.200714 |       2 | INSERT INTO t1 VALUES (1),(2)    | ODBC           |
| 00:00:00.000530 |      38 | SELECT start_ time,user_ host,qu | ODBC           |
+-----------------+---------+----------------------------------+----------+
39 rows in set (Elapsed: 00:00:00.00)

gbase> INSERT INTO t1 SELECT * FROM t1;
Query OK, 2 rows affected
gbase> UPDATE t1 SET i = 3;
Query OK, 4 rows affected
gbase> DELETE FROM t1;
Query OK, 4 rows affected
gbase> SELECT start_ time,user_ host,query_ time,rows, LEFT(sql_text, 30),
conn_ type FROM gbase.audit_ log;
```

```
+---------------------+--------------------------------+
| start_ time         | user_ host                     |
+---------------------+--------------------------------+
| 2021-06-16 11:17:18 | root[root] @ localhost []      |
| 2021-06-16 11:18:09 | root[root] @ localhost []      |
| 2021-06-16 11:18:21 | root[root] @    [192.168.146.24] |
| 2021-06-16 11:18:21 | root[root] @    [192.168.146.24] |
| 2021-06-16 11:18:21 | root[root] @    [192.168.146.24] |
| 2021-06-16 11:18:20 | root[root] @ localhost []      |
| 2021-06-16 11:18:29 | root[root] @    [192.168.146.24] |
| 2021-06-16 11:18:29 | root[root] @    [192.168.146.24] |
| 2021-06-16 11:18:29 | root[root] @    [192.168.146.24] |
| 2021-06-16 11:18:28 | root[root] @ localhost []      |
+---------------------+--------------------------------+


+-----------------+------+--------------------------------+-----------+
| query_ time     | rows | LEFT(sql_text, 30)             | conn_ type |
+-----------------+------+--------------------------------+-----------+
| 00:00:00.003119 |    0 | truncate self table gbase.audi | ODBC      |
| 00:00:02.090188 |    2 | INSERT INTO t1 SELECT * FROM t | ODBC      |
| 00:00:00.000095 |    0 | commit                         | CAPI      |
| 00:00:00.000272 |    0 | set autocommit=1               | CAPI      |
| 00:00:00.000416 |    0 | set autocommit=1               | CAPI      |
| 00:00:01.108768 |    4 | UPDATE t1 SET i = 3            | ODBC      |
| 00:00:00.000057 |    0 | commit                         | CAPI      |
| 00:00:00.000198 |    0 | set autocommit=1               | CAPI      |
| 00:00:00.000899 |    0 | set autocommit=1               | CAPI      |
| 00:00:00.396723 |    4 | DELETE FROM t1                 | ODBC      |
+-----------------+------+--------------------------------+-----------+
10 rows in set (Elapsed: 00:00:00.00)
```

4.8 Backup Recovery Management

4.8.1 Backup Recovery Management

GBase 8a MPP Cluster provides a dedicated backup and recovery tool (gcrcman), which allows users to easily backup and recover data from the entire cluster. The backup and recovery tools for the cluster are automatically installed with the installation of the cluster, and this tool is installed in the installation directory $GCLUSTER_ Under BASE/server/bin.

Start a new cycle with a full backup. An incremental backup renews the last backup cycle, increasing it by one backup point.

This chapter mainly introduces how to use backup and recovery tools for data backup and recovery, including:

- Cluster level full backup: Backs up the current cluster's data in full to the specified backup directory (ensuring that the directory has been created).
- Cluster level incremental backup: Based on the full or incremental backup data in the specified backup directory, incrementally backup the data of the current cluster to that backup directory.
- Library level full backup: Back up all tables and normal views under a certain library.
- Library level incremental backup: Incremental backup of all tables under a certain library.
- Table level full backup: Backs up the full data of a table to the backup directory.
- Table level incremental backup: Incrementally backup the data of a table to the backup directory.
- Cluster level recovery: Restores the specified backup data in the backup directory to the current cluster.
- Library level recovery: Restores all tables and normal views under a library.
- Table level recovery: Restores the data of a single table in the backup directory to the current database.
- View backup data: After data backup, check which data has been backed up.
- Delete backup data: Delete the backup data specified by the user.
- Delete junk data: Due to abnormalities or user interruptions, residual junk backup data can be deleted by users through tools.

be careful

- When executing the gcrcman. py command, it must be the dbauser specified when installing the database.
- The backup and recovery operation needs to be performed on the coordinator node, and the network connections of each node in the cluster are normal during the backup and recovery operation.
- To execute the gcrcman. py command, it is necessary to ensure that the Expect package is installed on the operating system.
- During the backup and recovery process, the path specified by the parameter path in gcrcman. py must exist on each node of the cluster. The cluster installation user dbauser has read and write permissions on this path, and the directory exists throughout the execution process.
- Do not include $GCLUSTER_ BASE、$GBASE_ BASE、$GCWARE_ The three directories and their subdirectories of BASE are set as paths.
- During the backup and recovery process, the passwords of the operating system and database users cannot be changed.
- When performing backup and recovery operations, the topology of the cluster during recovery needs to be the same as that during backup, including management nodes, data nodes, data distribution information, etc.
- Whether it is multiple session connections or a single session connection, only one gcrcman. py program can be run at a time.
- Currently, data backup and recovery for a single VC is not supported.

The backup objects supported by GCluster 8a MPP Cluster in Table -450 are explained as follows

(√: supported; ×： Not supported)

Database objects	Cluster level	Library level	Table level
Table metadata	√	√	√
Table data	√	√	√
Table index	√	√	√
stored procedure	√	×	×
view	√	√	×
UDF	×	×	×
Mirror Table	√	×	×
Tablespace	√	√	×

4.8.2 Grammar format

```
$ python $GCLUSTER_ BASE/server/bin/ gcrcman.py [options] <-d|--path BACKUP_ PATH>
```

Table -451 Options Options Description

Parameter Name	explain
-h,--help	Help information for displaying command parameters
-V,--version	Used to display version information of gcrcman
-d BACKUP_ PATH,--path=BACK UP_ PATH	Used to set the storage path for backup data. The path must be absolute, and the use of '~' is supported in the path. There must be a path specified by the path parameter below each node, and this path requires write permission. Cannot add $GCLUSTER_ BASE or $GBASE_ BASE or $GCWARE_ The three directories and subdirectories of BASE are set as paths.
-e COMMAND,--execut e=COMMAND	Specify the backup or restore command to be executed, and automatically exit after execution. Refer to the table for specific commands. When using the abbreviated parameter - e, it is recommended to use a space between - e and the following parameters to improve readability.
-P HOST_ PASSWD,--ospasswo rd=HOST_ PASSWD	Specifies the operating system user password of the database administrator. The default is gbase user and cannot be specified. Please fill in the password of the operating system user gbase here.
-p DATABASE_ PASSWD,--dbpasswo rd=DATABASE_ PASSWD	Specify the database user password of the database administrator. Some operations of the backup and recovery tool require access to the database, so it is necessary to specify the database user password. The default database user is gbase and cannot be specified.
-r PARALLEL_ LEVEL,--parallel=PA RALLEL_ LEVEL	Used to set the parallelism of backup and recovery tool execution, with a value range of [1128] and a default value of 4.
-D,--disk_ space_ estimate	Used to set whether to perform spatial estimation. If this parameter is present, no spatial estimation will be performed. By default, this parameter is not used, which means space estimation is performed during backup.
-c,--checksum_	Used to set whether to perform data verification. If this

Parameter Name	explain
database	parameter is present, DC checksum verification is not performed. By default, this parameter is not used, which means data validation is performed during backup.
-C,--checksum_ backup_ data	Used to set whether to perform verification of backup data. By default, this parameter is not used to verify backup data.
-t SECOND,--timeout=SECOND	Used to set the duration of waiting for read and write transactions. During backup, it is necessary to wait for no write transactions in the cluster. When restoring backup data, it is necessary to wait for no read and write transactions in the cluster before proceeding with subsequent operations. The unit of this parameter is seconds, and the value range is [03600]. If this parameter is not specified, the default value is 300 seconds. If it times out, an error will be reported to exit, indicating that the backup or recovery has failed. The parameter value is 0, indicating infinite waiting.
-R，--dop	The parallelism of performing batch backups or restores, with a range of [1128] and a default of 4

Table -452 Backup and Recovery Command Description

Parameter Name	explain
show backup	Display backup information
show backup tables [cycle_id]	Display bulk backup data
backup level <0|1>	Cluster level backup
backup database [vcname.]<dbname> level <0|1>	Database level backup
backup table [vcname.]<dbname.tablename> level <0|1>	Table level backup
backup tables <table.list> level <0|1>	Batch table level backup Within the same backup cycle, the table. list corresponding to full backup and incremental backup must be consistent. During full backup, new tables can be added, while incremental backup cannot add new

Parameter Name	explain
	tables.
recover [<cycle_id> [point_id]]	Instance level recovery
recover database [vcname.]<dbname> [<cycle_id> [point_id]]	Database level recovery
recover [force] table [vcname.]<dbname.tablename> [<cycle_id> [point_id]]	Table level recovery
recover [force] tables [<cycle_id> [point_id]]	Batch Table Level Recovery
delete <cycle_ id | last>	Delete backup data
cleanup	Clear invalid backup data
quit	sign out
help	display help information

Table -453 Description of parameters involved in backup and recovery commands

Parameter Name	explain
level <0|1>	Set the backup level, where 0 represents full backup and 1 represents incremental backup.
vcname	VC name
dbname	Database name
tablename	Table Name
cycle_ id	Latest backup point
point_ id	Specify backup point

4.8.3 **Execution mode**

Gcrcman supports two execution modes, namely interactive mode and command-line mode:

1. In interactive mode, enter the gcrcman>prompt to perform backup and recovery operations. The specific commands are as follows:

```
# su - gbase
$ gcrcman.py -d/home/gbase/backuptest/
gcrcman> show backup
```

```
cycle   point   level   time
0       0       0       2017-06-10 21:20:52
1       0       0       2017-06-10 21:20:54
1       1       1       2017-06-10 21:20:56
1       2       1       2017-06-10 21:20:59
1       3       1       2017-06-10 21:21:01
1       4       1       2017-06-10 21:21:03
1       5       1       2017-06-10 21:21:05
2       0       0       2017-06-10 21:21:09
3       0       0       2017-06-10 21:21:23
4       0       0       2017-06-10 21:21:40
5       0       0       2017-06-10 21:21:43
6       0       0       2017-06-10 21:21:58
7       0       0       2017-06-10 21:22:00
```

2. Command line mode, using the - e parameter, after gcrcman starts, only execute the command specified by - e, and automatically exit after execution. The specific commands are as follows:

```
# su - gbase
$ cd $GCLUSTER_ BASE/server/bin
$ python gcrcman.py -d /home/gbase/backuptest -e "show backup"
cycle   point   level   time
0       0       0       2017-06-10 21:20:52
1       0       0       2017-06-10 21:20:54
1       1       1       2017-06-10 21:20:56
1       2       1       2017-06-10 21:20:59
1       3       1       2017-06-10 21:21:01
1       4       1       2017-06-10 21:21:03
1       5       1       2017-06-10 21:21:05
2       0       0       2017-06-10 21:21:09
3       0       0       2017-06-10 21:21:23
4       0       0       2017-06-10 21:21:40
5       0       0       2017-06-10 21:21:43
6       0       0       2017-06-10 21:21:58
7       0       0       2017-06-10 21:22:00
```

4.8.4 Cluster backup

be careful

- When backing up, it is necessary for the cluster to be in a normal state.
- Due to the interaction between gcrcman and gcware, backup operations need to be performed on the coordinator where gcware services are installed.
- The cluster topology cannot be changed. The topology structure includes coordinator nodes, data nodes, distribution, and the corresponding relationships between various segments of distribution and datanodes.
- During backup, due to the gcrcman tool running on the node to obtain the current time as the backup point's backup time, it is important for the cluster to keep the time of each node consistent in order to ensure continuous backup on different nodes.
- During instance level backup, it is necessary to use the cluster installation user dbauser (default gbase), execute gcadmin switchmode readonly, set the cluster to read-only state, and after the backup is completed, execute gcadmin switchmode normal to restore the cluster to normal state. Otherwise, error information will be reported, such as:

```
gcrcman>backup level 0

11.13 09:52:43    [ERROR]:    (GBA-02BR-0065) The gcware not in

'READONLY' mode, please switch this mode by hand!
```

- During backup, the program, along with all users and passwords in the database, is backed up at the same time. Therefore, during recovery, it is also recommended that customers record the default root and gbase user passwords in the cluster before backup to avoid forgetting them during recovery.
- When a cluster undergoes expansion or contraction operations, the structure of the cluster will change. Therefore, the original backup records will become invalid and the recovery operation cannot be completed. The correct approach is to perform a new backup operation after expanding or shrinking the cluster, so that data can be restored through backup records.
- When backing up, the data is encrypted using a key, and the key needs to be saved. When restoring the data, the decryption process requires the original key

4.8.4.1 Cluster level backup of cluster data

Function Description

Run backup level [0 | 1] in interactive mode or command mode to complete data backup.

When performing backup operations, it is necessary to ensure that the cluster is in

READONLY mode.

Firstly, use the gcadmin switchmode readonly command to set the cluster to READONLY mode. This command can only be executed once on a node where gcware is deployed.

After execution, start the cluster backup command to proceed with the backup operation.

Grammar format

```
backup level < 0 | 1 >
```

Example

Step 1: View the cluster status:

```
$ gcadmin
CLUSTER STATE:              ACTIVE


=============================================================
======
|                  GBASE COORDINATOR CLUSTER INFORMATION
|
=============================================================
======
|     NodeName     |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |         0         |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |         0         |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |         0         |
-------------------------------------------------------------
=============================================================
|                  GBASE VIRTUAL CLUSTER INFORMATION                    |
=============================================================
|       VcName       | DistributionId |                comment                |
-----------------------------------------------------------
|         vc1          |             1              | comment message for vc1 |
-----------------------------------------------------------
|         vc2          |             2              | comment message for vc2 |
-----------------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node
```

```
$ gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

============================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                    |
============================================================
|     VcName       | DistributionId |               comment               |
------------------------------------------------------------
|       vc1          |            1             | comment message for vc1 |
------------------------------------------------------------
============================================================
=======
|                           VIRTUAL CLUSTER DATA NODE INFORMATION
|
============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-------------------------------------------------------------------
| node1    |172.168.83.11|          1            |OPEN |     OPEN     |       0       |
-------------------------------------------------------------------
| node2    |172.168.83.12|          1            |OPEN |     OPEN     |       0       |
-------------------------------------------------------------------

2 data node

$ gcadmin showcluster vc vc2
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

============================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                    |
============================================================
|     VcName       | DistributionId |               comment               |
------------------------------------------------------------
|       vc2          |            2             | comment message for vc2 |
------------------------------------------------------------
============================================================
=======
|                           VIRTUAL CLUSTER DATA NODE INFORMATION
|
============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
```

```
-------------------------------------------------------------------
| node1   |172.168.83.13|          2          |OPEN |     OPEN     |      0      |
-------------------------------------------------------------------
| node2   |172.168.83.14|          2          |OPEN |     OPEN     |      0      |
-------------------------------------------------------------------

2 data node
Step 2: Set the status of all VCs in the cluster to readonly:
$ gcadmin switchmode readonly vc vc1

========== switch cluster mode...
   switch pre mode:                    [NORMAL]
   switch mode to                      [READONLY]
   switch after mode:                  [READONLY]
$ gcadmin switchmode readonly vc vc2

========== switch cluster mode...
   switch pre mode:                    [NORMAL]
   switch mode to                      [READONLY]
   switch after mode:                  [READONLY]
$ gcadmin
CLUSTER STATE:              ACTIVE

============================================================
=====
|                  GBASE COORDINATOR CLUSTER INFORMATION
|
============================================================
=====
|     NodeName     |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |   OPEN   |    OPEN    |       0       |
-------------------------------------------------------------
============================================================
|                  GBASE VIRTUAL CLUSTER INFORMATION                  |
============================================================
|      VcName      | DistributionId |              comment              |
-------------------------------------------------------------
|       vc1        |          1          | comment message for vc1 |
-------------------------------------------------------------
```

```
|        vc2          |            2             | comment message for vc2 |
-------------------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node

$ gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    READONLY

=============================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                    |
=============================================================
|     VcName       | DistributionId |                comment                |
-------------------------------------------------------------
|       vc1          |            1             | comment message for vc1 |
-------------------------------------------------------------
=============================================================
=======
|                          VIRTUAL CLUSTER DATA NODE INFORMATION
|
=============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-----------------------------------------------------------------
| node1    |172.168.83.11|          1            |OPEN |      OPEN     |       0       |
-----------------------------------------------------------------
| node2    |172.168.83.12|          1            |OPEN |      OPEN     |       0       |
-----------------------------------------------------------------

2 data node

$ gcadmin showcluster vc vc2
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    READONLY

=============================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                    |
=============================================================
|       VcName       | DistributionId |                comment                |
-------------------------------------------------------------
|        vc2          |            2             | comment message for vc2 |
-------------------------------------------------------------
```

```
============================================================
=======
|                        VIRTUAL CLUSTER DATA NODE INFORMATION
|
============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
---------------------------------------------------------------------
| node1    |172.168.83.13|          2          |OPEN |     OPEN     |      0      |
---------------------------------------------------------------------
| node2    |172.168.83.14|          2          |OPEN |     OPEN     |      0      |
---------------------------------------------------------------------

2 data node
Using interactive mode for full backup:
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d /home/gbase/backup -P
gbasedba  – p ******
gcrcman>backup level 0
08.07 17:24:34    BackUp start
-------------------------------------------
08.07 17:24:34    node (172.168.83.11)    backup begin
08.07 17:24:34    node (172.168.83.12)    backup begin
08.07 17:24:35    node (172.168.83.13)    backup begin
08.07 17:24:35    node (172.168.83.14)    backup begin
08.07 17:25:16    node (172.168.83.11)    backup success
08.07 17:25:16    node (172.168.83.12)    backup success
08.07 17:25:16    node (172.168.83.13)    backup success
08.07 17:25:16    node (172.168.83.14)    backup success
-------------------------------------------
08.07 17:25:17    BackUp end
Using interactive mode for provisioning:
gcrcman>backup level 1
08.07 19:28:53    check cluster topology begin
08.07 19:28:53    node (172.168.83.11)    check topology begin
08.07 19:28:56    node (172.168.83.11)    check topology success
08.07 19:28:56    check cluster topology end
08.07 19:28:56    BackUp start
-------------------------------------------
08.07 19:28:56    node (172.168.83.11)    backup begin
08.07 19:28:56    node (172.168.83.12)    backup begin
08.07 19:28:56    node (172.168.83.13)    backup begin
08.07 19:28:57    node (172.168.83.14)    backup begin
08.07 19:29:31    node (172.168.83.11)    backup success
08.07 19:29:31    node (172.168.83.12)    backup success
```

```
08.07 19:29:31    node (172.168.83.13)    backup success
08.07 19:29:31    node (172.168.83.14)    backup success
-----------------------------------------------
08.07 19:29:31    BackUp end
After the backup is completed, set the cluster state to normal:
$ gcadmin switchmode normal vc vc1

========== switch cluster mode...
   switch pre mode:                [READONLY]
   switch mode to                  [NORMAL]
   switch after mode:              [NORMAL]
$ gcadmin switchmode normal vc vc2

========== switch cluster mode...
   switch pre mode:                [READONLY]
   switch mode to                  [NORMAL]
   switch after mode:              [NORMAL]
```

4.8.4.2 Library level backup

Function Description

Fully/incrementally backup all tables in a library in the GBase 8a MPP Cluster database to the specified path.

Grammar format

```
backup database [vc_name].< database_ name>level < 0 | 1 >
```

Table -454 Options Option Description

Parameter Name	explain
vc_ name	The virtual cluster name of the database to be restored
Database_ name	Database name to be restored
Level < 0 | 1 >	Backed up

Example

Example 1: Using command line mode for library level backup.

```
Check cluster status:
$ gcadmin
CLUSTER STATE:              ACTIVE
```

```
============================================================
=====
|                  GBASE COORDINATOR CLUSTER INFORMATION
|
============================================================
=====
|     NodeName     |     IpAddress     | gcware | gcluster | DataState |
------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |         0         |
------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |         0         |
------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |         0         |
------------------------------------------------------------
============================================================
|                  GBASE VIRTUAL CLUSTER INFORMATION                    |
============================================================
|       VcName       | DistributionId |                comment                |
------------------------------------------------------------
|         vc1          |             1             | comment message for vc1 |
------------------------------------------------------------
|         vc2          |             2             | comment message for vc2 |
------------------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node

$ gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

============================================================
|                  GBASE VIRTUAL CLUSTER INFORMATION                    |
============================================================
|     VcName       | DistributionId |                comment                |
------------------------------------------------------------
|       vc1          |             1             | comment message for vc1 |
------------------------------------------------------------
============================================================
=======
|                          VIRTUAL CLUSTER DATA NODE INFORMATION
|
============================================================
```

```
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
---------------------------------------------------------------------
| node1    |172.168.83.11|          1          |OPEN |     OPEN     |      0      |
---------------------------------------------------------------------
| node2    |172.168.83.12|          1          |OPEN |     OPEN     |      0      |
---------------------------------------------------------------------

2 data node

$ gcadmin showcluster vc vc2
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

============================================================
|                  GBASE VIRTUAL CLUSTER INFORMATION                    |
============================================================
|       VcName       | DistributionId |               comment               |
------------------------------------------------------------
|         vc2          |            2             | comment message for vc2 |
------------------------------------------------------------
============================================================
=======
|                          VIRTUAL CLUSTER DATA NODE INFORMATION
|
============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
---------------------------------------------------------------------
| node1    |172.168.83.13|          2          |OPEN |     OPEN     |      0      |
---------------------------------------------------------------------
| node2    |172.168.83.14|          2          |OPEN |     OPEN     |      0      |
---------------------------------------------------------------------

2 data node
Using command line mode for full backup:
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d /home/gbase/backupD -P
gbasedba     – p ******     -e "backup database vc1.demo level 0"
08.08 14:10:32    BackUp database vc1.demo start
-----------------------------------------------
08.08 14:10:32    node (172.168.83.11)    backup database begin
08.08 14:10:32    node (172.168.83.12)    backup database begin
08.08 14:10:32    node (172.168.83.13)    backup database begin
08.08 14:10:32    node (172.168.83.14)    backup database begin
```

```
08.08 14:10:53    node (172.168.83.11)    backup database success
08.08 14:10:53    node (172.168.83.12)    backup database success
08.08 14:10:53    node (172.168.83.13)    backup database success
08.08 14:10:53    node (172.168.83.14)    backup database success
---------------------------------------------
08.08 14:10:53    BackUp database vc1.demo end

Using command line mode for backup:
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d /home/gbase/backupD -P
gbasedba -e "backup database vc1.demo level 1"
08.08 14:58:21    BackUp database vc1.demo start
---------------------------------------------
08.08 14:58:21    node (172.168.83.11)    backup database begin
08.08 14:58:21    node (172.168.83.12)    backup database begin
08.08 14:58:21    node (172.168.83.13)    backup database begin
08.08 14:58:21    node (172.168.83.14)    backup database begin
08.08 14:58:44    node (172.168.83.11)    backup database success
08.08 14:58:44    node (172.168.83.12)    backup database success
08.08 14:58:44    node (172.168.83.13)    backup database success
08.08 14:58:44    node (172.168.83.14)    backup database success
---------------------------------------------
08.08 14:58:44    BackUp database vc1.demo end
```

4.8.4.3 Table level backup

Function Description

Fully backup a table in the GBase 8a MPP Cluster database to the specified path.

Grammar format

```
backup table [vcname].< dbname>.< tablename> level < 0 | 1 >
```

Example

Example 1: Using command line mode for table level backup.

```
Check cluster status:
$ gcadmin
CLUSTER STATE:              ACTIVE


=============================================================
=====
|                  GBASE COORDINATOR CLUSTER INFORMATION
|
=============================================================
```

```
=====
|     NodeName     |     IpAddress     | gcware | gcluster | DataState |
-------------------------------------------------------------
| coordinator1 | 172.168.83.11 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator2 | 172.168.83.12 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
| coordinator3 | 172.168.83.13 |    OPEN    |     OPEN     |        0        |
-------------------------------------------------------------
============================================================
|                   GBASE VIRTUAL CLUSTER INFORMATION                    |
============================================================
|       VcName       | DistributionId |                comment                |
-----------------------------------------------------------
|         vc1          |            1             | comment message for vc1 |
-----------------------------------------------------------
|         vc2          |            2             | comment message for vc2 |
-----------------------------------------------------------

2 virtual cluster: vc1, vc2
3 coordinator node
0 free data node

$ gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

============================================================
|                   GBASE VIRTUAL CLUSTER INFORMATION                    |
============================================================
|     VcName       | DistributionId |                comment                |
-----------------------------------------------------------
|       vc1          |            1             | comment message for vc1 |
-----------------------------------------------------------
============================================================
=======
|                            VIRTUAL CLUSTER DATA NODE INFORMATION
|
============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-------------------------------------------------------------------
| node1    |172.168.83.11|          1            |OPEN |      OPEN     |       0       |
-------------------------------------------------------------------
```

```
| node2    |172.168.83.12|          1          |OPEN |     OPEN     |       0       |
-------------------------------------------------------------------

2 data node

$ gcadmin showcluster vc vc2
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    NORMAL

============================================================
|                  GBASE VIRTUAL CLUSTER INFORMATION                    |
============================================================
|       VcName       | DistributionId |                comment                |
------------------------------------------------------------
|         vc2          |            2            | comment message for vc2 |
------------------------------------------------------------
============================================================
=======
|                          VIRTUAL CLUSTER DATA NODE INFORMATION
|
============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-------------------------------------------------------------------
| node1    |172.168.83.13|          2          |OPEN |     OPEN     |       0       |
-------------------------------------------------------------------
| node2    |172.168.83.14|          2          |OPEN |     OPEN     |       0       |
-------------------------------------------------------------------

2 data node
Using command line mode for full backup:
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d /home/gbase/backupT -P
gbasedba      – p ******     -e "backup table vc1.demo.t level 0"
08.08 15:00:35    BackUp table vc1.demo.t start
---------------------------------------------
08.08 15:00:36    node (172.168.83.11)    backup table begin
08.08 15:00:36    node (172.168.83.12)    backup table begin
08.08 15:00:36    node (172.168.83.13)    backup table begin
08.08 15:00:36    node (172.168.83.14)    backup table begin
08.08 15:00:59    node (172.168.83.11)    backup table success
08.08 15:00:59    node (172.168.83.12)    backup table success
08.08 15:00:59    node (172.168.83.13)    backup table success
08.08 15:00:59    node (172.168.83.14)    backup table success
---------------------------------------------
```

```
08.08 15:00:59    BackUp table vc1.demo.t end
Using command line mode for backup:
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d /home/gbase/backupT -P
gbasedba     – p ******     -e "backup table vc1.demo.t level 1"
08.08 15:02:49    check cluster topology begin
08.08 15:02:49    node (172.168.83.11)    check topology begin
08.08 15:02:51    node (172.168.83.11)    check topology success
08.08 15:02:52    check cluster topology end
08.08 15:02:52    BackUp table vc1.demo.t start
---------------------------------------------
08.08 15:02:52    node (172.168.83.11)    backup table begin
08.08 15:02:52    node (172.168.83.12)    backup table begin
08.08 15:02:52    node (172.168.83.13)    backup table begin
08.08 15:02:52    node (172.168.83.14)    backup table begin
08.08 15:03:16    node (172.168.83.11)    backup table success
08.08 15:03:16    node (172.168.83.12)    backup table success
08.08 15:03:16    node (172.168.83.13)    backup table success
08.08 15:03:16    node (172.168.83.14)    backup table success
---------------------------------------------
08.08 15:03:16    BackUp table vc1.demo.t end
```

4.8.4.4 Batch table level backup

Function Description

Batch backup the table specified in the configuration file table.list.

be careful:

- During full backup, new tables can be added to table.list, but during incremental backup, new tables cannot be added.
- When backing up, it is necessary to ensure that the service and data states of the cluster are normal and there is no feventlog.
- A backup directory needs to be created in advance on each node, and the backup user has read and write permissions.
- Only express engine tables can be backed up.
- If there are write operations on the backed up table, the backup program will wait.

Grammar format

```
backup tables.< table.list> level < 0 | 1 >
```

Table. list is the name of the configuration file and can be named arbitrarily. When writing, an absolute path needs to be written.

The table. list content is the name of the table that needs to be backed up, in the format vcname.db.table, with one table name per row. If there is a default VC, vcname can be omitted.

Example

The test library includes tables t1, t2, and t3

Create backup directory/home/gbase/backup on each node of the cluster

Create the/home/gbase/table. list file:

vcname000001.test.t1

vcname000001.test.t2

vcname000001.test.t3

```
[ gbase@rhel73-1    ~]$ gcrcman.py -P "******" -p "******" -d
"/home/gbase/backup" -R 8
gcrcman>backup tables /home/gbase/table.list level 0
==========BackUpMultiTable==========
========
distinct table list ['vcname000001.test.t1', 'vcname000001.test.t2',
'vcname000001.test.t3']
==========BackUp ParallelProcessTable default parallel 8 tablenum
3==========
==========BackUpOneTable==========
=========
backup table vcname000001.test.t1 level 0
==========BackUpOneTable==========
=========
backup table vcname000001.test.t2 level 0
==========BackUpOneTable==========
=========
backup table vcname000001.test.t3 level 0
05.19 13:48:11    BackUp table vcname000001.test.t1 start
-------------------------------------------
05.19 13:48:12    node (192.168.146.40)    backup table begin
05.19 13:48:12    node (192.168.146.41)    backup table begin
05.19 13:48:12    node (192.168.146.42)    backup table begin
```

```
05.19 13:48:16    BackUp table vcname000001.test.t2 start
-----------------------------------------------
05.19 13:48:16    node (192.168.146.40)    backup table begin
05.19 13:48:16    node (192.168.146.41)    backup table begin
05.19 13:48:16    node (192.168.146.42)    backup table begin
05.19 13:48:18    BackUp table vcname000001.test.t3 start
-----------------------------------------------
05.19 13:48:18    node (192.168.146.40)    backup table begin
05.19 13:48:18    node (192.168.146.41)    backup table begin
05.19 13:48:18    node (192.168.146.42)    backup table begin
05.19 13:48:38    node (192.168.146.40)    backup table success
05.19 13:48:38    node (192.168.146.41)    backup table success
05.19 13:48:38    node (192.168.146.42)    backup table success
-----------------------------------------------
05.19 13:48:38    BackUp table vcname000001.test.t1 end
05.19 13:48:40    node (192.168.146.40)    backup table success
05.19 13:48:40    node (192.168.146.41)    backup table success
05.19 13:48:40    node (192.168.146.42)    backup table success
-----------------------------------------------
05.19 13:48:40    BackUp table vcname000001.test.t2 end
05.19 13:48:41    node (192.168.146.40)    backup table success
05.19 13:48:41    node (192.168.146.41)    backup table success
05.19 13:48:41    node (192.168.146.42)    backup table success
-----------------------------------------------
05.19 13:48:41    BackUp table vcname000001.test.t3 end
====================ParallelProcessTable
End============================
=========================SaveBackUpInfo=======================
==========
gcrcman>backup tables /home/gbase/table.list level 1
=========================BackUpMultiTable=====================
========
distinct table list ['vcname000001.test.t1', 'vcname000001.test.t2',
'vcname000001.test.t3']
==============BackUp ParallelProcessTable default parallel 8 tablenum
```

```
3==============
==========BackUpOneTable==========
=========
backup table vcname000001.test.t1 level 1
==========BackUpOneTable==========
=========
backup table vcname000001.test.t2 level 1
==========BackUpOneTable==========
=========
backup table vcname000001.test.t3 level 1
05.19 13:50:41    BackUp table vcname000001.test.t1 start
-----------------------------------------
05.19 13:50:42    node (192.168.146.40)    backup table begin
05.19 13:50:42    node (192.168.146.41)    backup table begin
05.19 13:50:42    node (192.168.146.42)    backup table begin
05.19 13:50:44    BackUp table vcname000001.test.t2 start
-----------------------------------------
05.19 13:50:45    BackUp table vcname000001.test.t3 start
-----------------------------------------
05.19 13:50:45    node (192.168.146.40)    backup table begin
05.19 13:50:45    node (192.168.146.41)    backup table begin
05.19 13:50:45    node (192.168.146.40)    backup table begin
05.19 13:50:45    node (192.168.146.41)    backup table begin
05.19 13:50:46    node (192.168.146.42)    backup table begin
05.19 13:50:46    node (192.168.146.42)    backup table begin
05.19 13:51:11    node (192.168.146.40)    backup table success
05.19 13:51:11    node (192.168.146.41)    backup table success
05.19 13:51:11    node (192.168.146.42)    backup table success
-----------------------------------------
05.19 13:51:11    BackUp table vcname000001.test.t1 end
05.19 13:51:12    node (192.168.146.40)    backup table success
05.19 13:51:12    node (192.168.146.41)    backup table success
05.19 13:51:12    node (192.168.146.42)    backup table success
-----------------------------------------
05.19 13:51:12    BackUp table vcname000001.test.t3 end
```

```
05.19 13:51:12    node (192.168.146.40)    backup table success
05.19 13:51:12    node (192.168.146.41)    backup table success
05.19 13:51:12    node (192.168.146.42)    backup table success
-------------------------------------------
05.19 13:51:12    BackUp table vcname000001.test.t2 end
==========ParallelProcessTable
End==========
==========SaveBackUpInfo==========
==========
```

Note: If the parameter - R is not added, the default parallelism is 4.

4.8.4.5 Remote backup

To ensure the security of backup files, a dedicated file server is selected to store backup records.

Users set the shared path on the file server, and then mount the shared path in nfs mode on the main shard node of the cluster through the mount command. Backup records can be saved on a remote file server.

The advantages of doing so are as follows:

- Backup record files can be stored on remote file servers;
- The security of backup record files can be guaranteed;

If the entire cluster contains a total of 2 main shard nodes, we will select 2 file servers to store backup records on each main shard.

The main sharding node and mounting path are as follows:

Table -455 Main sharding nodes and mounting paths

Main sharding IP	Path on the main shard (to be used as a mount point in the future)
192.168.103.209	/home/gbase/backuptest
192.168.103.211	/home/gbase/backuptest

The file server and sharing are as follows:

Table -456 File Server and Share Path

File Server IP	Shared Path
192.168.103.222	/home/safegroup1
192.168.103.223	/home/safegroup2

Below, we will introduce to users how to share files using nfs, as follows:

On each file server, use root to check if the rpm package for nfs is installed. If not, please

install it yourself.

```
# rpm -qa | grep nfs
nfs-utils-lib-1.1.5-4.el6.x86_ sixty-four
nfs-utils-1.2.3-15.el6.x86_ sixty-four
nfs4-acl-tools-0.3.3-5.el6.x86_ sixty-four
```

On each file server, if there is no cluster installation user dbauser, create a gbase user and set a password (consistent with the gbase user password of the cluster operating system).

```
# useradd gbase
# passwd gbase
Changing password for user gbase.
New UNIX password:
BAD PASSWORD: it is too short
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

On each file server, switch to cluster installation user dbauser and create a shared path.

```
# su – gbase
--The shared path created by each file server
$ mkdir safegroup1
```

On each file server, use the cluster to install user dbauser and set permissions and user groups for the shared path.

```
$ chmod -R 777 /home/gbase/safegroup1
$ chown -R gbase:gbase /home/gbase/safegroup1
```

On each file server, use the root user to set the nfs service to start on its own after startup.

```
# chkconfig nfs on
```

Using root user, set sharing on 192.168.103.222.

```
# vi /etc/exports
/home/gbase/safegroup1 *(rw)
~
```

Using root user, set sharing on 192.168.103.223.

```
# vi /etc/exports
/home/safegroup2 *(rw)
~
```

On each file server, use root to start the nfs service.

```
# service nfs start
Start NFS service: [OK]
Turn off NFS quota: [OK]
Start NFS daemon: [OK]
Start NFS mount: [OK]
```

On the main partition of 192.168.103.209, use root user to perform mount mount sharing.

```
# mount -t nfs -o rw 192.168.103.222:/home/safegroup1 /home/gbase/backuptest
```

On the main partition of 192.168.103.211, use root user to perform mount mount sharing.

```
# mount -t nfs -o rw 192.168.103.223:/home/safegroup2 /home/gbase/backuptest
```

After mounting on the main shard and creating the same path as the mount point on the main shard on other cluster nodes, users can perform backup operations using the local cluster backup method. After the operation is completed, the backup records are saved on their respective file servers.

 **explain**

- If the mount is unmounted, the backup will fail.
- For example, to cancel the mount mount on the main partition 192.168.103.211, you can use the root user to execute the umount command, as follows:

```
# umount /home/gbase/backuptest
```

- If the main sharding machine of the cluster restarts, use the root user on each main sharding machine to execute the mount command and re mount after the restart. The reference commands are as follows:

```
# mount -t nfs -o rw 192.168.103.223: /home/safegroup2
/home/gbase/backuptest
```

4.8.5 Cluster recovery

be careful

- The cluster state must be normal during recovery.
- The recovery operation needs to be performed on the coordinator.
- The cluster topology cannot be changed. The topology structure includes: Coordinator node, Datanode node, distribution, and the corresponding relationship between each segment of distribution and datanode.
- The cluster topology before recovery must be consistent with the backup.
- To use the instance level data recovery command, it is necessary to ensure that the cluster is in RECOVERY mode: use the cluster installation user dbauser (default gbase), execute gcadmin switchmode recovery, set the cluster to the recovery state, and then start the cluster recovery command to perform the recovery operation.

4.8.5.1 Cluster level backup data recovery

Function Description

Restore cluster level backup data. When performing backup and recovery operations, it is necessary to ensure that the cluster is in RECOVERY mode.

Firstly, use the gcadmin switchmode recovery command to set the cluster to RECOVERY mode. This command can only be executed once on a node where gcware is deployed.

After the recovery execution is completed, it is necessary to manually execute gcadmin switchmode normal to set the cluster status to normal before the cluster can be used normally.

be careful

For cluster level recovery, strict equivalence of cluster topology is required. This includes the VC information of the cluster, the number of coordinator nodes, the number of data nodes, the correspondence between each node and sharding, and the distribution ID. After the user reinstalls the cluster, even if the user guarantees the number of coordinators, datanodes, and the relationship between datanodes and distribution, it may not be restored due to changes in the distribution ID.

Grammar format

```
recover [<cycle_id> [point_id]]
```

Table -457 Options Options Description

Parameter Name	explain
cycle_ id	ID of the backup cycle
point_ id	The ID of the backup point

The recover command has the following three forms:

- Recover: Restore the cluster to the latest backup point of the latest cycle.
- recover cycle_ ID: Restore the cluster to the specified cycle_ The latest backup point within the ID.
- recover cycle_ id point_ ID: Restore the cluster to the specified cycle_ The specified backup point within the id_ id。

Example

```
View cluster status:
$ gcadmin showcluster vc vc1
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    RECOVERY


============================================================
|                GBASE VIRTUAL CLUSTER INFORMATION                    |
============================================================
|     VcName       | DistributionId |                comment                |
------------------------------------------------------------
|       vc1          |            1             | comment message for vc1 |
------------------------------------------------------------
============================================================
=======
|                    VIRTUAL CLUSTER DATA NODE INFORMATION
|
============================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-------------------------------------------------------------------
| node1    |172.168.83.11|          1            |OPEN |     OPEN     |      0      |
-------------------------------------------------------------------
| node2    |172.168.83.12|          1            |OPEN |     OPEN     |      0      |
-------------------------------------------------------------------


2 data node


$ gcadmin showcluster vc vc2
```

```
CLUSTER STATE:                ACTIVE
VIRTUAL CLUSTER MODE:    RECOVERY


=========================================================
|              GBASE VIRTUAL CLUSTER INFORMATION                    |
=========================================================
| VcName       | DistributionId |               comment               |
---------------------------------------------------------
|   vc2          |            2             | comment message for vc2 |
---------------------------------------------------------
=========================================================
=======
|                   VIRTUAL CLUSTER DATA NODE INFORMATION
|
=========================================================
=======
|NodeName|    IpAddress    |DistributionId|gnode|syncserver|DataState|
-------------------------------------------------------------------
| node1    |172.168.83.13|          2            |OPEN |     OPEN     |      0      |
-------------------------------------------------------------------
| node2    |172.168.83.14|          2            |OPEN |     OPEN     |      0      |
-------------------------------------------------------------------
```

2 data node

Restore the backup data of the cluster to the nearest backup point of the latest cycle:

```
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d /home/gbase/backup -P
gbasedba – p ****** -e "recover"
08.10 18:14:13    check cluster topology begin
08.10 18:14:13    node (172.168.83.11)    check topology begin
08.10 18:14:16    node (172.168.83.11)    check topology success
08.10 18:14:16    check cluster topology end
08.10 18:14:16    check BackUp start
---------------------------------------------
08.10 18:14:16    node (172.168.83.11)    check backup begin
08.10 18:14:16    node (172.168.83.12)    check backup begin
08.10 18:14:16    node (172.168.83.13)    check backup begin
08.10 18:14:16    node (172.168.83.14)    check backup begin
08.10 18:14:41    node (172.168.83.11)    check backup success
08.10 18:14:41    node (172.168.83.12)    check backup success
08.10 18:14:41    node (172.168.83.13)    check backup success
08.10 18:14:41    node (172.168.83.14)    check backup success
---------------------------------------------
08.10 18:14:41    check BackUp success
```

```
08.10 18:14:41    Recover start
--------------------------------------------
08.10 18:14:43    node (172.168.83.11)    Recover begin
08.10 18:14:43    node (172.168.83.12)    Recover begin
08.10 18:14:43    node (172.168.83.13)    Recover begin
08.10 18:14:43    node (172.168.83.14)    Recover begin
08.10 18:15:28    node (172.168.83.11)    Recover success
08.10 18:15:28    node (172.168.83.12)    Recover success
08.10 18:15:28    node (172.168.83.13)    Recover success
08.10 18:15:28    node (172.168.83.14)    Recover success
--------------------------------------------
08.10 18:15:28    Recover success
```

Restore the backup data of the cluster and restore the latest backup point of the second backup cycle of the cluster:

```
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d /home/gbase/backup -P
gbasedba -e "recover 1"
08.10 18:41:32    check cluster topology begin
08.10 18:41:32    node (172.168.83.11)    check topology begin
08.10 18:41:35    node (172.168.83.11)    check topology success
08.10 18:41:35    check cluster topology end
08.10 18:41:35    check BackUp start
--------------------------------------------
08.10 18:41:35    node (172.168.83.11)    check backup begin
08.10 18:41:35    node (172.168.83.12)    check backup begin
08.10 18:41:35    node (172.168.83.13)    check backup begin
08.10 18:41:35    node (172.168.83.14)    check backup begin
08.10 18:42:05    node (172.168.83.11)    check backup success
08.10 18:42:05    node (172.168.83.12)    check backup success
08.10 18:42:05    node (172.168.83.13)    check backup success
08.10 18:42:05    node (172.168.83.14)    check backup success
--------------------------------------------
08.10 18:42:05    check BackUp success
08.10 18:42:05    Recover start
--------------------------------------------
08.10 18:42:07    node (172.168.83.11)    Recover begin
08.10 18:42:07    node (172.168.83.12)    Recover begin
08.10 18:42:07    node (172.168.83.13)    Recover begin
08.10 18:42:07    node (172.168.83.14)    Recover begin
08.10 18:42:46    node (172.168.83.11)    Recover success
08.10 18:42:46    node (172.168.83.12)    Recover success
08.10 18:42:46    node (172.168.83.13)    Recover success
08.10 18:42:46    node (172.168.83.14)    Recover success
--------------------------------------------
08.10 18:42:46    Recover success
```

Restore the backup data of the cluster and restore the second backup point of the third backup cycle of the cluster:

```
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d /home/gbase/backup -P gbasedba -e "recover 2 1"
08.10 18:48:55    check cluster topology begin
08.10 18:48:55    node (172.168.83.11)    check topology begin
08.10 18:48:58    node (172.168.83.11)    check topology success
08.10 18:48:58    check cluster topology end
08.10 18:48:58    check BackUp start
--------------------------------------------
08.10 18:48:58    node (172.168.83.11)    check backup begin
08.10 18:48:58    node (172.168.83.12)    check backup begin
08.10 18:48:58    node (172.168.83.13)    check backup begin
08.10 18:48:58    node (172.168.83.14)    check backup begin
08.10 18:49:17    node (172.168.83.11)    check backup success
08.10 18:49:17    node (172.168.83.12)    check backup success
08.10 18:49:17    node (172.168.83.13)    check backup success
08.10 18:49:17    node (172.168.83.14)    check backup success
--------------------------------------------
08.10 18:49:17    check BackUp success
08.10 18:49:17    Recover start
--------------------------------------------
08.10 18:49:19    node (172.168.83.11)    Recover begin
08.10 18:49:19    node (172.168.83.12)    Recover begin
08.10 18:49:19    node (172.168.83.13)    Recover begin
08.10 18:49:19    node (172.168.83.14)    Recover begin
08.10 18:49:51    node (172.168.83.11)    Recover success
08.10 18:49:51    node (172.168.83.12)    Recover success
08.10 18:49:51    node (172.168.83.13)    Recover success
08.10 18:49:51    node (172.168.83.14)    Recover success
--------------------------------------------
08.10 18:49:52    Recover success
```

After the backup is completed, set the cluster state to normal:

```
$ gcadmin switchmode normal vc vc1

========== switch cluster mode...
   switch pre mode:                [READONLY]
   switch mode to                  [NORMAL]
   switch after mode:              [NORMAL]
$ gcadmin switchmode normal vc vc2

========== switch cluster mode...
   switch pre mode:                [READONLY]
   switch mode to                  [NORMAL]
```

```
switch after mode:                    [NORMAL]
```

4.8.5.2 Database level backup data recovery

Function Description

Restore the backup data from the specified library in the backup directory to the GBase8a MPP Cluster database.

be careful

- You need to delete the database before restoring it.
- After the database recovery is successful, the following operations need to be manually performed to use the recovered database tables: refresh all tables in the database: refresh table table table names;
  Or
  restart the cluster service

Grammar format

```
recover database [vcname.] <database_ name> [<cycle_id> [point_id]]
```

Table -458 Options Option Description

Parameter Name	explain
vc_ name	The virtual cluster name of the database to be restored
database_ name	Database name to be restored
cycle_ id	ID of the backup cycle
point_ id	The ID of the backup point

The recover command has the following three forms:

- Recover database vcname.dbname: Restores the database to the latest backup point of the latest cycle.
- recover database vcname.dbname cycle_ ID: Restore the database to the specified cycle_ The latest backup point within the ID.
- recover database vcname.dbname cycle_ id point_ ID: Restore the database to the specified cycle_ The specified backup point within the id_ id。

Example

Example 1: Restore database level backup data in the database.

```
Delete the database to be restored:
$ gccli -uroot -e"drop database vc1.demo"
```

```
 gccli -uroot -e"select * from vc1.demo.t"
ERROR 1146 (42S02) at line 1: Table 'vc1.demo.t' doesn't exist
```

After the backup is completed, set the cluster state to normal:

```
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d
/home/gbase/backupD -P gbasedba -e "recover database vc1.demo"
08.10 21:11:10    check database BackUp start
--------------------------------------------
08.10 21:11:10    node (172.168.83.11)    check database backup begin
08.10 21:11:10    node (172.168.83.12)    check database backup begin
08.10 21:11:10    node (172.168.83.13)    check database backup begin
08.10 21:11:10    node (172.168.83.14)    check database backup begin
08.10 21:11:32    node (172.168.83.11)    check database backup success
08.10 21:11:32    node (172.168.83.12)    check database backup success
08.10 21:11:32    node (172.168.83.13)    check database backup success
08.10 21:11:32    node (172.168.83.14)    check database backup success
--------------------------------------------
08.10 21:11:32    check database BackUp success
08.10 21:11:32    recover prepare start
--------------------------------------------
08.10 21:11:32    node (172.168.83.11)    recover prepare begin
08.10 21:11:32    node (172.168.83.12)    recover prepare begin
08.10 21:11:32    node (172.168.83.13)    recover prepare begin
08.10 21:11:32    node (172.168.83.14)    recover prepare begin
08.10 21:11:38    node (172.168.83.11)    recover prepare success
08.10 21:11:38    node (172.168.83.12)    recover prepare success
08.10 21:11:38    node (172.168.83.13)    recover prepare success
08.10 21:11:38    node (172.168.83.14)    recover prepare success
--------------------------------------------
08.10 21:11:38    recover prepare finish
08.10 21:11:38    recreate database vc1.demo start
08.10 21:11:38    recreate database vc1.demo end
08.10 21:11:38    recreate database vc1.demo tables start
08.10 21:11:38    node (172.168.83.11)    recreate database tables begin
08.10 21:11:41    node (172.168.83.11)    recreate database tables success
08.10 21:11:41    recreate database vc1.demo tables end
08.10 21:11:41    Recover database vc1.demo start
```

```
---------------------------------------------
08.10 21:11:41    node (172.168.83.11)    Recover database begin
08.10 21:11:41    node (172.168.83.12)    Recover database begin
08.10 21:11:41    node (172.168.83.13)    Recover database begin
08.10 21:11:41    node (172.168.83.14)    Recover database begin
08.10 21:11:58    node (172.168.83.11)    Recover database success
08.10 21:11:58    node (172.168.83.12)    Recover database success
08.10 21:11:58    node (172.168.83.13)    Recover database success
08.10 21:11:58    node (172.168.83.14)    Recover database success
---------------------------------------------
08.10 21:11:58    Recover database vc1.demo success, please refresh it!


Restart cluster service:
$ gcluster_ services all restart
Stopping gcrecover :                                       [   OK   ]
Stopping gcluster :                                        [   OK   ]
Stopping gbase :                                           [   OK   ]
Stopping syncserver :                                      [   OK   ]
Starting gbase :                                           [   OK   ]
Starting syncserver :                                      [   OK   ]
Starting gcluster :                                        [   OK   ]
Starting gcrecover :                                       [   OK   ]
```

4.8.5.3 Table level backup data recovery

Function Description

Restore the backup data of the specified table in the backup directory to the GBase8a MPP Cluster database.

be careful

- You need to delete the table before restoring it.
- After the recovery is completed, a refresh table operation needs to be performed on the recovered table in order for it to be used normally. For example: refresh table vcname.dbname.tbname;

Grammar format

```
recover table <vc_ name>.< database_ name>.< table_ name> [<cycle_id>
[point_id]]
```

Table -459 Options Options Description

Parameter Name	explain
vc_ name	The virtual cluster name of the database to be restored
database_ name	The database name to which the table to be restored belongs
table_ name	Table name to be restored
cycle_ id	ID of the backup cycle
point_ id	The ID of the backup point

The recover command has the following three forms:

- Recover table vcname.dbname.tbname: Restores the table to the latest backup point of the latest cycle.
- recover table vcname.dbname.tbname cycle_ Id: Restore the table to the specified cycle_ The latest backup point within the ID.
- recover table vcname.dbname.tbname cycle_ id point_ Id: Restore the table to the specified cycle_ The specified backup point within the id_ id。

Example

Example 1: Restoring table level backup data in a database.

```
Delete the table to be restored:
$ gccli -uroot -e"drop table vc1.demo.t"
$
Recovery Table:
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d
/home/gbase/backupT -P gbasedba -e "recover table vc1.demo.t"
08.10 22:32:42    check Table topology start
-------------------------------------------
08.10 22:32:42    node (172.168.83.11)    check table topology begin
08.10 22:32:44    node (172.168.83.11)    check table topology success
-------------------------------------------
08.10 22:32:44    check table topology success
08.10 22:32:44    check Table BackUp start
-------------------------------------------
```

```
08.10 22:32:44    node (172.168.83.11)    check table backup begin
08.10 22:32:44    node (172.168.83.12)    check table backup begin
08.10 22:32:44    node (172.168.83.13)    check table backup begin
08.10 22:32:44    node (172.168.83.14)    check table backup begin
08.10 22:33:02    node (172.168.83.11)    check table backup success
08.10 22:33:02    node (172.168.83.12)    check table backup success
08.10 22:33:02    node (172.168.83.13)    check table backup success
08.10 22:33:02    node (172.168.83.14)    check table backup success
-----------------------------------------------
08.10 22:33:02    check table BackUp success
08.10 22:33:02    refresh table vc1.demo.t start
08.10 22:33:02    node (172.168.83.11)    recreate table begin
08.10 22:33:08    node (172.168.83.11)    recreate table success
08.10 22:33:08    refresh table vc1.demo.t end
08.10 22:33:09    Recover table vc1.demo.t start
-----------------------------------------------
08.10 22:33:09    node (172.168.83.11)    Recover table begin
08.10 22:33:09    node (172.168.83.12)    Recover table begin
08.10 22:33:09    node (172.168.83.13)    Recover table begin
08.10 22:33:09    node (172.168.83.14)    Recover table begin
08.10 22:33:33    node (172.168.83.11)    Recover table success
08.10 22:33:33    node (172.168.83.12)    Recover table success
08.10 22:33:33    node (172.168.83.13)    Recover table success
08.10 22:33:33    node (172.168.83.14)    Recover table success
-----------------------------------------------
08.10 22:33:33    Recover table vc1.demo.t success, please refresh it!
```

Swipe message table information and query:

```
$ gccli -uroot vc1.demo    -e"refresh table vc1.demo.t"
$ gccli -uroot vc1.demo    -e"select * from vc1.demo.t"
+------+-----------+-------------------+
| a    | b         | c                 |
+------+-----------+-------------------+
|    1 | aa        | ccccccc           |
|    2 | aadad     | ccchjfghjfcccc    |
|    3 | aaasdexfg | cccfhgfgjfghjcccc |
```

```
+------+-----------+-------------------+
```

4.8.5.4 Batch table level backup data recovery

Function Description

Restore the backup data of the tables in the backup directory to the GBase8a MPP Cluster database.

Grammar format

```
recover [force] tables [<cycle_id> [point_id]]
```

Recover tables: Restore to the last backup point of the maximum backup cycle

recover tables cycle_ Id: Restore to the last backup point of the specified backup cycle

recover tables cycle_ id point_ Id: Restore to the backup point of the specified backup cycle

be careful:

When restoring, it is necessary to ensure that no table with the same name as the table to be restored exists in the library

The cluster topology and table distribution rules during recovery are the same as those during backup

The table being restored will prevent DDL and DML operations on that table

Example

```
[ gbase@rhel73-1    ~]$ gcrcman.py -P "******" -p "******" -d
"/home/gbase/backup" -R 8
gcrcman>recover force tables 0 0
==========RecoverMultiTable==========
==========
recover to cycle 0 point 0
table_ list ['vcname000001.test.t1', 'vcname000001.test.t2', 'vcname000001.test.t3']
==========Recover ParallelProcessTable default parallel 8 tablenum
3==========
==========RecoverOneTable_ CallBack
vcname000001.test.t1==========
```

```
==============RecoverOneTable_ CallBack
vcname000001.test.t2==============
==============RecoverOneTable_ CallBack
vcname000001.test.t3==============
05.19 13:57:16    check Table topology start
-----------------------------------------------
05.19 13:57:16    node (192.168.146.40)    check table topology begin
05.19 13:57:17    check Table topology start
-----------------------------------------------
05.19 13:57:17    node (192.168.146.40)    check table topology begin
05.19 13:57:17    check Table topology start
-----------------------------------------------
05.19 13:57:17    node (192.168.146.40)    check table topology begin
05.19 13:57:19    node (192.168.146.40)    check table topology success
-----------------------------------------------
05.19 13:57:19    check table topology success
05.19 13:57:19    check Table BackUp start
-----------------------------------------------
05.19 13:57:19    node (192.168.146.40)    check table backup begin
05.19 13:57:19    node (192.168.146.41)    check table backup begin
05.19 13:57:19    node (192.168.146.42)    check table backup begin
05.19 13:57:22    node (192.168.146.40)    check table topology success
-----------------------------------------------
05.19 13:57:22    check table topology success
05.19 13:57:22    check Table BackUp start
-----------------------------------------------
05.19 13:57:22    node (192.168.146.40)    check table backup begin
05.19 13:57:22    node (192.168.146.41)    check table backup begin
05.19 13:57:22    node (192.168.146.42)    check table backup begin
05.19 13:57:22    node (192.168.146.40)    check table topology success
-----------------------------------------------
05.19 13:57:22    check table topology success
05.19 13:57:22    check Table BackUp start
-----------------------------------------------
05.19 13:57:22    node (192.168.146.40)    check table backup begin
```

```
05.19 13:57:22    node (192.168.146.41)    check table backup begin
05.19 13:57:22    node (192.168.146.42)    check table backup begin
05.19 13:57:35    node (192.168.146.40)    check table backup success
05.19 13:57:35    node (192.168.146.41)    check table backup success
05.19 13:57:35    node (192.168.146.42)    check table backup success
-----------------------------------------------
05.19 13:57:35    check table BackUp success
05.19 13:57:36    refresh table vcname000001.test.t1 start
05.19 13:57:36    node (192.168.146.40)    recreate table begin
05.19 13:57:37    node (192.168.146.40)    check table backup success
05.19 13:57:37    node (192.168.146.41)    check table backup success
05.19 13:57:37    node (192.168.146.42)    check table backup success
-----------------------------------------------
05.19 13:57:37    check table BackUp success
05.19 13:57:38    refresh table vcname000001.test.t2 start
05.19 13:57:38    node (192.168.146.40)    recreate table begin
05.19 13:57:38    node (192.168.146.40)    check table backup success
05.19 13:57:38    node (192.168.146.41)    check table backup success
05.19 13:57:38    node (192.168.146.42)    check table backup success
-----------------------------------------------
05.19 13:57:38    check table BackUp success
05.19 13:57:38    refresh table vcname000001.test.t3 start
05.19 13:57:38    node (192.168.146.40)    recreate table begin
05.19 13:57:40    node (192.168.146.40)    recreate table success
05.19 13:57:40    refresh table vcname000001.test.t1 end
05.19 13:57:40    Recover table vcname000001.test.t1 start
-----------------------------------------------
05.19 13:57:40    node (192.168.146.40)    Recover table begin
05.19 13:57:40    node (192.168.146.41)    Recover table begin
05.19 13:57:40    node (192.168.146.42)    Recover table begin
05.19 13:57:41    node (192.168.146.40)    recreate table success
05.19 13:57:41    refresh table vcname000001.test.t2 end
05.19 13:57:42    Recover table vcname000001.test.t2 start
-----------------------------------------------
05.19 13:57:42    node (192.168.146.40)    Recover table begin
```

```
05.19 13:57:42    node (192.168.146.41)    Recover table begin
05.19 13:57:42    node (192.168.146.42)    Recover table begin
05.19 13:57:43    node (192.168.146.40)    recreate table success
05.19 13:57:43    refresh table vcname000001.test.t3 end
05.19 13:57:44    Recover table vcname000001.test.t3 start
---------------------------------------------
05.19 13:57:44    node (192.168.146.40)    Recover table begin
05.19 13:57:44    node (192.168.146.41)    Recover table begin
05.19 13:57:44    node (192.168.146.42)    Recover table begin
05.19 13:58:02    node (192.168.146.40)    Recover table success
05.19 13:58:02    node (192.168.146.41)    Recover table success
05.19 13:58:02    node (192.168.146.42)    Recover table success
---------------------------------------------
05.19 13:58:02    Recover table vcname000001.test.t1 success, please refresh it!
05.19 13:58:03    node (192.168.146.40)    Recover table success
05.19 13:58:03    node (192.168.146.41)    Recover table success
05.19 13:58:03    node (192.168.146.42)    Recover table success
---------------------------------------------
05.19 13:58:03    Recover table vcname000001.test.t2 success, please refresh it!
05.19 13:58:03    node (192.168.146.40)    Recover table success
05.19 13:58:03    node (192.168.146.41)    Recover table success
05.19 13:58:03    node (192.168.146.42)    Recover table success
---------------------------------------------
05.19 13:58:03    Recover table vcname000001.test.t3 success, please refresh it!
====================ParallelProcessTable
End============================
```

4.8.6 View backup records

Function Description

This command can view the backup information of the cluster, including the backup cycle number (sorted from 0), backup point number (sorted from 0), backup type (full backup or incremental backup), and backup date and time.

Through the following example, users can view the backup records of the cluster database, so that users, especially the database administrator, can understand the backup of the cluster in a timely and effective manner.

 **explain**

If an error occurs during the backup process:

- If the backup records of each node in the cluster are inconsistent, some invalid backup records will be generated, and such records will not be displayed.
- If the password is incorrect due to user password modification, the record will be displayed.

Grammar format

```
show backup
```

Table -460 Explanation of Querying Backup Records

Column Name	explain
cycle	Backup cycle number
point	Backup point number
level	Backup level
time	Execute backup time

Example

Example 1: Viewing backup records in interactive mode.

```
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d /home/gbase/backup
-P gbasedba
gcrcman>show backup
                    cyclepointleveltime
      0002020-08-07 17:24:34
      0112020-08-07 19:28:56
      0212020-08-08 13:55:22
```

Example 2: Command line mode to view backup records.

```
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d /home/gbase/backup
-P gbasedba -e "show backup"
                    cyclepointleveltime
      0002020-08-07 17:24:34
0     1     1     2020-08-07 19:28:56
      0212020-08-08 13:55:22
      1002020-08-08 15:08:51
      1112020-08-08 15:10:55
```

Example 3: Command line mode to view batch table backup records.

```
[ gbase@rhel73-1    ~]$ gcrcman.py -P "******" -p "******" -d
"/home/gbase/backup" -R 8
gcrcman>show backup tables
cycle     point     level     time
0          0          0          2022-05-19 13:47:34
0          1          1          2022-05-19 13:50:17
----------table list----------
vcname000001.test.t1
vcname000001.test.t2
vcname000001.test.t3
```

4.8.7 Delete cluster backup records

Function Description

Delete the backed up data. Due to the fact that in practical applications, the "full increase full increase..." mode is generally adopted, which involves first performing a full backup, followed by several additional backups, followed by full backup, followed by several additional backups, and sequentially cycling the backup operation process. After a period of accumulated operations during the backup process, a large number of outdated backup records will accumulate. Therefore, users can use this command to clear the backup records.

be careful

- If there is only one "full increment" backup record, this command cannot be used to delete the backup record. To ensure the security of backup data, backup and recovery tools will refuse to delete the data from the last backup cycle.
- When deleting backup points, each group of main sharding nodes deletes a certain cycle or the last backup point according to the command requirements.
- The rule for deleting backup records is to delete the corresponding backup record files in the path of the main shard node. If the path specified by gcrcman. py in other nodes has been copied from the path of the primary sharding node, all backup files in the path of these non primary sharding node clusters will be deleted.

Grammar format

```
delete <cycle_ id | last>
```

Table -461 Options Options Description

Parameter Name	explain
cycle_ id	Delete all backup records within a cycle. After executing this option, the deletion cycle will be cycle_ Backup records of all backup points for ID.
last	Delete the last backup point in the backup record. After executing this option, the backup record of the last backup point in the last cycle will be deleted.

Example

Example 1: Delete the backup record of the last backup point in the last cycle.

```
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d
/home/gbase/backupT -P gbasedba -e "show backup"
                    cyclepointleveltime
0    0    0    2020-08-08 15:00:36
     0112020-08-08 15:02:52
     1002020-08-10 23:23:20
     1112020-08-10 23:24:32
1    2    1    2020-08-11 00:46:37
     2002020-08-11 00:49:26
     2112020-08-11 00:51:01
     3002020-08-11 00:53:23
3    1    1    2020-08-11 00:57:35
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d
/home/gbase/backupT -P gbasedba -e "delete last"
08.11 00:59:48  Delete BackUp Points start
--------------------------------------------
08.11 00:59:48  node (172.168.83.11)  delete begin
08.11 00:59:48  node (172.168.83.12)  delete begin
08.11 00:59:48  node (172.168.83.13)  delete begin
08.11 00:59:48  node (172.168.83.14)  delete begin
08.11 00:59:59  node (172.168.83.11)  delete success
08.11 00:59:59  node (172.168.83.12)  delete success
08.11 00:59:59  node (172.168.83.13)  delete success
08.11 00:59:59  node (172.168.83.14)  delete success
--------------------------------------------
08.11 00:59:59  Delete BackUp Points end
```

```
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d
/home/gbase/backupT -P gbasedba -e "show backup"
cycle point level  time
        0002020-08-08 15:00:36
        0112020-08-08 15:02:52
        1002020-08-10 23:23:20
1       1       1       2020-08-10 23:24:32
        1212020-08-11 00:46:37
        2002020-08-11 00:49:26
        2112020-08-11 00:51:01
        3002020-08-11 00:53:23
```

Example 2: Deleting a cycle_ All backup records with id 2.

```
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d
/home/gbase/backupT -P gbasedba -e "show backup"
                        cyclepointleveltime
        0002020-08-08 15:00:36
0       1       1       2020-08-08 15:02:52
        1002020-08-10 23:23:20
        1112020-08-10 23:24:32
        1212020-08-11 00:46:37
2       0       0       2020-08-11 00:49:26
        2112020-08-11 00:51:01
        3002020-08-11 00:53:23
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d
/home/gbase/backupT -P gbasedba -e "delete 2"
08.11 01:09:38    Delete BackUp Points start
---------------------------------------------
08.11 01:09:38    node (172.168.83.11)    delete begin
08.11 01:09:38    node (172.168.83.12)    delete begin
08.11 01:09:38    node (172.168.83.13)    delete begin
08.11 01:09:38    node (172.168.83.14)    delete begin
08.11 01:09:48    node (172.168.83.11)    delete success
08.11 01:09:48    node (172.168.83.12)    delete success
08.11 01:09:48    node (172.168.83.13)    delete success
```

```
08.11 01:09:48    node (172.168.83.14)    delete success
--------------------------------------------
08.11 01:09:48    Delete BackUp Points end
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d
/home/gbase/backupT -P gbasedba -e "show backup"
                    cyclepointleveltime
        0002020-08-08 15:00:36
0       1       1       2020-08-08 15:02:52
        1002020-08-10 23:23:20
        1112020-08-10 23:24:32
        1212020-08-11 00:46:37
3       0       0       2020-08-11 00:53:23
```

4.8.8 Clear invalid backup data from the cluster

Function Description

Clear invalid backup data in the cluster. In the actual backup process, due to various unexpected reasons, the backup records in the main sharding node may be inconsistent. When there is invalid backup data, gcrcman needs to delete it, otherwise subsequent operations such as backup and recovery cannot be performed. Execute cleanup to delete junk backup data. If there is no invalid backup data, the message 'no need to rollback' will be prompted when performing cleanup. The cleanup command cannot be used to delete successfully backed up data. To delete data files in the backup directory, you need to manually delete them using the Linux rm command.

Grammar format

```
cleanup
```

Example

Example 1: Clearing invalid backup data.

```
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d
/home/gbase/backupT -P gbasedba -e "show backup"
                    cyclepointleveltime
0       0       0       2020-08-08 15:00:36
        0112020-08-08 15:02:52
```

```
        1002020-08-10 23:23:20
        1112020-08-10 23:24:32
1       2       1       2020-08-11 00:46:37
        3002020-08-11 00:53:23
$ python $GCLUSTER_ BASE/server/bin/gcrcman.py -d
/home/gbase/backupT -P gbasedba -e "cleanup"
08.11 01:49:00    cleanup rubbish file or dir start
---------------------------------------------
08.11 01:49:00    node (172.168.83.11)    cleanup rubbish file or dir begin
08.11 01:49:00    node (172.168.83.12)    cleanup rubbish file or dir begin
08.11 01:49:00    node (172.168.83.13)    cleanup rubbish file or dir begin
08.11 01:49:00    node (172.168.83.14)    cleanup rubbish file or dir begin
08.11 01:49:15    node (172.168.83.11)    cleanup rubbish file or dir success
08.11 01:49:15    node (172.168.83.12)    cleanup rubbish file or dir success
08.11 01:49:15    node (172.168.83.13)    cleanup rubbish file or dir success
08.11 01:49:15    node (172.168.83.14)    cleanup rubbish file or dir success
---------------------------------------------
08.11 01:49:15    cleanup rubbish file or dir end
no need to rollback
```

4.9 security management

4.9.1 User and Permission Management

4.9.1.1 user management

4.9.1.1.1 Create a new user

Operation scenario

Create a new GBase 8a MPP Cluster database user account.

prerequisite

The user performing this operation must have global CREATE USER permission.

Grammar format

Create a new GBase 8a MPP Cluster database user account using the CREATE USER statement, with the following syntax:

```
CREATE USER user [IDENTIFIED BY [PASSWORD] [password]]
```

Table -462 Parameter Description

Parameter Name	explain
USER *user*	Specify the account name to create. The *user* supports the following writing methods: 1. Any host: *user* @ '%' 2. Local machine: *user* @'localhost ' 3. Network segment: *user* @'192.168.0.0 ' 4. IP address: *user* @'192.168.10.6 ' 5. Equivalent to user @ '%': *user* 6. The user length supports 128 characters
[IDENTIFIED BY [PASSWORD] [*password*]]	You can assign a password to the account through the optional Identify By statement. It should be noted that the PASSWORD keyword can be omitted when the password is set to plain text format. *Password* is the account password

explain

- For each unauthorized account, initially only login to the database is allowed.
- Create user specified user@host When host is a specific fixed and unique string, the account login list hosts cannot be specified. If it must be specified, the hosts should be the same as host. Please refer to 4.9.5.3 Account Restricted Host List for specific syntax.

Example

Example 1

Create an admin user.

```
gbase> CREATE USER admin IDENTIFIED BY 'admin';
Query OK, 0 rows affected
```

```
gbase> EXIT
Bye

$ gccli -uadmin –padmin
GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights
Reserved.
```

Example 2

Log in as super user root and create a user user1.

```
$ gccli -uroot -p
Enter password:

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights
Reserved.

gbase> CREATE USER user1;
Query OK, 0 rows affected
```

4.9.1.1.2 Modify User Name

Operation scenario

Rename an existing GBase 8a MPP Cluster user.

prerequisite

Modifying a username must have global CREATE USER permission.

Grammar format

Use the RENAME USER statement to modify the user name, with the following syntax format:

```
RENAME USER old_ user TO new_ user
```

explain

- After modifying the username, the permissions remain unchanged.

Example

```
gbase> RENAME USER user1 to user2;
```

```
Query OK, 0 rows affecte
```

4.9.1.1.3 delete user

Operation scenario

Delete existing GBase 8a MPP Cluster users.

prerequisite

Deleting user information must have global CREATE USER permission.

Grammar format

Use the DROP USER statement to delete a user, with the following syntax format:

```
DROP USER user;
```

explain

- DROP USER will not automatically close any open user sessions. More precisely, when there are sessions opened using this user, deleting this user does not affect the access rights of these opened sessions until the session is closed

Example

Example 1

Remove the admin user.

```
gbase> DROP USER admin;

Query OK, 0 rows affected
```

4.9.1.1.4 Change User Password

Grammar format

Use SET PASSWORD to modify the user password, with the following syntax format:

```
SET PASSWORD [FOR user] = PASSWORD('newpassword')
```

Table -463 Parameter Description

Parameter Name	explain
FOR *user*	Specify the account name to modify the password. If omitted, it indicates modifying the password of the current login cluster account.
PASSWORD('*newpassword*')	Specify a new password for the account.

Example

Example 1

Set the password for the account currently logged into the cluster.

```
$ gccli –uroot

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights Reserved

gbase> SET PASSWORD = PASSWORD('admin');

Query OK, 0 rows affected

gbase> EXIT

Bye
```

Log in to the database again with the new password:

```
$ gccli -uroot -p

Enter password:

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights Reserved.
```

Example 2

Set the password for the specified login cluster user.

```
$ gccli -uroot --nice_ time_ format -p

Enter password:

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights Reserved.

gbase> SET PASSWORD FOR admin = PASSWORD('adminnew');
```

```
Query OK, 0 rows affected

gbase> EXIT

Bye

$ gccli -uadmin --nice_ time_ format -p

Enter password:

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights
Reserved.
```

4.9.1.2 User Group Management

The user group function is to form a group of users with the same permissions. Users only need to modify the permissions of the group, and the permissions of all users under the group will be changed, making it easier to manage users with the same permissions.

4.9.1.2.1 Create User Group

Grammar format

Create a user group using the CREATE ROLE statement, with the following syntax format:

```
CREATE ROLE [IF NOT EXISTS] role [, role ] .....
```

Example

```
gbase> create role test2,test3;

Query OK, 0 rows affected (Elapsed: 00:00:00.04)
```

4.9.1.2.2 Delete User Group

Grammar format

Delete user groups in the cluster. The syntax format is as follows:

```
DROP ROLE [IF EXISTS] role [, role ] ...
```

4.9.1.3 **Permission management**

Operation scenario

Administrators plan the responsibilities of different database users and assign them corresponding operation permissions to ensure the safe operation of the database.

prerequisite

To use GRANT or REVOKE, users must have GRANT OPTION permission, whi ch can be granted or revoked.

Operating Steps

The system administrator grants and revokes permissions to users through GRANT and REVOKE statements. Please refer to the syntax format for using GRANT and REVOKE statements.

Example

Example 1: Using super dbauser root to create a user_ General user, who has the authority to perform SELECT operations.

```
$ gccli -uroot -p
Enter password:
GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.    All Rights Reserved.
gbase> CREATE USER user_ general;
Query OK, 0 rows affected
gbase> SET PASSWORD FOR user_ general = PASSWORD('H% 897_@m');
Query OK, 0 rows affected
For user_ General users are only granted SELECT permission. ** Represents all database objects, such as tables, views, stored procedures, etc.

gbase> GRANT SELECT ON *.* TO user_ general;
Query OK, 0 rows affected
gbase> \q
Bye
Using user_ Log in to the database and verify that it has select permission. There is a test database and a t1 table, which were created in advance for the demonstration example.

$ gccli -uuser_ general -p
```

```
Enter password:

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.    All Rights
Reserved.

gbase> USE test;
Query OK, 0 rows affected

gbase> UPDATE t1 SET a = 11 WHERE a = 10;
ERROR 1142 (42000): UPDATE command denied to user 'user_
general'@'localhost' for table 't1'

gbase> DELETE FROM t1;
ERROR 1142 (42000): DELETE command denied to user 'user_
general'@'localhost' for table 't1'

gbase> SELECT * FROM t1;
    +------+
    | a    |
    +------+
    |    1 |
    |    2 |
    |    3 |
    |    4 |
    |    5 |
    |    6 |
    |    7 |
    |    8 |
    |    9 |
    |   10 |
    +------+
10 rows in set
```

Example 2: Using super dbauser root to create a user_ The admin user has the privileges of a super user, that is, full privileges.

```
$ gccli -uroot -p
Enter password:

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.    All Rights
Reserved.
gbase> CREATE USER user_ admin;
Query OK, 0 rows affected

gbase> SET PASSWORD FOR user_ admin = PASSWORD('H% 897_@m ');
```

```
Query OK, 0 rows affected
```

For user_ Admin user grants full permissions. ** Database objects representing all databases, such as tables, views, and stored procedures.

```
gbase> GRANT ALL ON *.* TO user_ admin;
Query OK, 0 rows affected
```

Using user_ The admin user logs in to the database and verifies that they have f ull permissions to perform operations such as SELECT, UPDATE, DELETE, CRE ATE USER, and DROP USER. Create the test database and t1 table in advance for the demonstration example.

```
$ gccli -uuser_ admin -p
Enter password:

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.    All Rights
Reserved.

gbase> USE test;
Query OK, 0 rows affected

gbase> SELECT * FROM t1;
    +------+
    | a    |
    +------+
    |    1 |
    |    2 |
    |    3 |
    |    4 |
    |    5 |
    |    6 |
    |    7 |
    |    8 |
    |    9 |
    |   10 |
    +------+
    10 rows in set

gbase> UPDATE t1 SET a = 11 WHERE a = 10;
Query OK, 1 row affected

gbase> SELECT * FROM t1;
    +------+
```

```
| a    |
+------+
|    1 |
|    2 |
|    3 |
|    4 |
|    5 |
|    6 |
|    7 |
|    8 |
|    9 |
|   11 |
+------+
10 rows in set

gbase> DELETE FROM t1 WHERE a >= 5;
Query OK, 6 rows affected

gbase> SELECT * FROM t1;
+------+
| a    |
+------+
|    1 |
|    2 |
|    3 |
|    4 |
+------+
4 rows in set
```

4.9.1.3.1 Using GRANT and REVOKE statements

Operation scenario

The GRANT and REVOKE statements allow system administrators to handle the granting and revocation of user permissions.

prerequisite

To use GRANT or REVOKE, users must have GRANT OPTION permission, which can be granted or revoked.

Grammar format

```
GRANT
        priv_ type [(column_list)]
        [, priv_type [(column_list)]] ...
        ON [object_type] priv_ level
         TO user IDENTIFIED BY [[PASSWORD] [password]]
          [WITH with_option ...]
object_ type:
        TABLE   | FUNCTION   | PROCEDURE
priv_ level:
            *   | *.*   | database_ name.*   |database_ name.table_ name
        | table_ name   | database_ name.routine_ name
```

```
REVOKE
            priv_ type [(column_list)]
              [, priv_type [(column_list)]] ...
            ON [object_type] priv_ level
              FROM user
REVOKE ALL PRIVILEGES, GRANT OPTION
FROM user
```

explain

For GRANT and REVOKE statements, priv_ Level can grant different levels of permissions:

- Global level: Global permissions are applied to all databases of a given server. These permissions are stored in the gbase. user table. Global permissions can only be granted and revoked through GRANT ALL ON *. *. * and REVOKE ALL ON *. *. *;
- Database level: Database permissions apply to all objects of a given database. These permissions are stored in the gbase.db and gbase.host tables. Only GRANT ALL ON VC can be used_ name.db_ Name. * and REVOKE ALL ON VC_ name.db_ Name. * Granting and revoking database permissions;
- Table level: Table permissions apply to all columns of a given table. These permissions are stored in gbase. tables_ Priv table. Only GRANT ALL ON vcname.db can be used_ name.tbl_ Name and REVOKE ALL ON vcname.db_ name.tbl_ Name grants and revokes table permissions;
- Column level: Column permissions are applied to the specified columns in the table. These permissions are stored in gbase. tables_ Priv table. Only GRANT SELECT (column), Insert (column), UPDATE (column) ON vcname.db can be used_ name.tb1_ Name and REVOKE SELECT (column), Insert (column), UPDATE (column) ON vcname.db_ name.tb1_ Name grants and revokes column permissions.

Example

Example 1: Granting select permission to column a in the t table to user admin.

```
gbase> CREATE TABLE t(a int,b varchar(40));
Query OK, 0 rows affected

gbase> INSERT INTO t VALUES (1,'test'),(2,'share');
Query OK, 2 rows affected
Records: 2  Duplicates: 0  Warnings: 0

gbase> GRANT SELECT(a) ON test.t TO admin;
Query OK, 0 rows affected

gbase> SELECT * FROM gbase.tables_ priv;
+-----------+----------+----------+---------------+-------------------+
| Host      | Db       | User     | Table_ name   | Grantor           |
+-----------+----------+----------+---------------+-------------------+
| %         | test     | admin    | t             | root@192.168.10.115 |
+-----------+----------+----------+---------------+-------------------+
+---------------------+------------+-------------+
| Timestamp           | Table_ priv | Column_ priv |
+---------------------+------------+-------------+
| 2013-10-18 14:52:33 |            | Select      |
```

```
+--------------------+------------+-------------+
```

Example 2: Retrieve the SELECT permission for column a in table t.

```
gbase> REVOKE SELECT(a) ON test.t FROM admin;
Query OK, 0 rows affected

gbase> SELECT * FROM gbase.tables_ priv;
Empty set
```

Example 3: Granting priority permissions to users, syntax: grant usage on *. *. * to user_ name with task_ priority priority_ value。

```
gbase> create user uer1 ;
Query OK, 0 rows affected

gbase> grant usage on *.*.* to uer1 with task_ priority    1;
Query OK, 0 rows affected
```

explain

- priority_ The value range is 0, 1, 2, and 3, corresponding to default priority, low priority, medium priority, and high priority respectively;
- Permission requirements: For users with grant permission, it is recommended to use dbauser: root.

be careful

After granting a specified permission to a user, if the object is deleted but the user's permission is not reclaimed, the user will have the permission to create a new object with the same name.

4.9.1.3.2 Permission level

Database permissions are divided into five categories: database object operation pe rmissions, data operation permissions, stored procedure and custom function execut ion permissions, data view permissions, and database (including user management) management permissions.

The following table shows the permission levels in GRANT and REVOKE, with priv in the syntax format_ Type is specified as any of the following.

Table -464 Permission Description

jurisdiction	significance
ALL [PRIVILEGES]	Set all simple permissions except GRANT OPTION
ALTER	Allow use of Alter Table
ALTER ROUTINE	Change or cancel stored subroutines
CREATE	Allow use of CREATE TABLE
CREATE ROUTINE	Create stored subroutines
CREATE TEMPORARY TABLES	Allow the use of CREATE TEMPORARY TABLES
CREATE USER	Allow the use of CREATE USER, DROP USER, RENAME USER, and REVOKE ALL [PRIVILEGES], GRANT OPTION
CREATE VIEW	Allow CREATE VIEW
DELETE	Allow DELETE
DROP	Allow DROP DATABASE and DROP TABLE to be used
DROP DATABASE	Allow DROP DATABASE
DROP TABLE	Allow DROP TABLE
DROP VIEW	Allow DROP VIEW
EXECUTE	Allow users to run stored subroutines
FILE	Allow SELECT FROM TABLE_ NAME INTO OUTFILE, LOAD DATA, etc
GRANT OPTION	Allow granting permissions
INDEX	Allow the use of CREATE INDEX and DROP INDEX
INSERT	Allow use of Insert
PROCESS	Allow the use of SHOW FULL PROCESS LISTS
RELOAD	Allow FLUSH
SELECT	Allow use of SELECT
SHOW DATABASES	Show DATABASES
SHOW VIEW	Allow SHOW CREATE VIEW
SHUTDOWN	Allow gbaseadmin shutdown
SUPER	Allow KILL and SET GLOBAL statements
UPDATE	Allow use of UPDATE
USAGE	Only used to connect and log in to the database, mainly for setting the with option section
UNMASK	Allow the use of UNMASK permission, and users with unmask permission can access actual data without being affected by desensitization rules
EVENT	Permission to create, modify, delete, and execute EVENT After reclaiming user EVENT permissions, the corresponding EVENT cannot continue to execute

4.9.1.3.3 Granting permissions to user groups

Operation scenario

User groups themselves have permissions, and they can be granted permissions like users.

Grammar format

```
GRANT SELECT ON DBNAME.TBNAME TO role_ name
```

Example

```
gbase> grant select on test.t to r1;
Query OK, 0 rows affected (Elapsed: 00:00:00.09)
```

4.9.1.3.4 Revoke permissions for user groups

Operation scenario

Reclaim database permissions for user groups.

Grammar format

```
REVOKE SELECT ON DBNAME.TBNAME FROM role_ name
```

Example

```
gbase> revoke select on test.t from r1;

Query OK, 0 rows affected (Elapsed: 00:00:00.07)

gbase> revoke all privileges on *.* from u2;

Query OK, 0 rows affected (Elapsed: 00:00:00.04)

gbase> revoke all privileges, grant option from u2;

Query OK, 0 rows affected (Elapsed: 00:00:00.03)

gbase> revoke all , grant option from u2;

Query OK, 0 rows affected (Elapsed: 00:00:00.02)
```

4.9.1.3.5 Granting User Group Permissions to Users

Operation scenario

Set the user group to which the user belongs, and the permissions of the user group will affect the user's permission authentication.

Grammar format

```
GRANT role [, role] … TO user [, user] [WITH ADMIN OPTION]
```

be careful

Using WITH ADMIN Option is to identify whether a user has the authority to grant user groups to other users.

4.9.1.3.6 Reclaim user group permissions

Operation scenario

Remove the user from the user group and the user will no longer have permissions in the user group.

Grammar format

```
REVOKE role [, role] …    FROM user [, user]……
```

4.9.2 Password Security Management

To ensure the consistency of password security management throughout the entire cluster environment, it is necessary to create a configuration file gbase for each coordinator node_ 8a_ Configuration files gbase for gcluster.cnf and each Data node_ 8a_ In gbase.cnf, configure the same values for the same variable.

The following read-only parameters can be modified in the following files for each gcluster node and gnode node (excluding gcware nodes) in the cluster:

Gcluster node:

$GCLUSTER_ BASE/config/gbase_ 8a_ gcluster.cnf

Gnode node:

$GBASE_ BASE/config/gbase_ 8a_ gbase.cnf

4.9.2.1 Password strength management

Users can configure password complexity and length requirements, which must be met when creating and modifying passwords.

Password complexity is affected by the read-only parameter password_ format_ Option control.

Password length is subject to read-only parameter password_ min_ Length control.

When the values of both variables are 0, password strength control is turned off.

Password strength control is effective for the following SQL types:

```
create user [user] identified by 'pass';
alter user [user] identified by 'pass';
set password = password('pass');
set password for [user] = password('pass');
```

The password strength control parameter is read-only and is defined as follows:

Table -465 password strength Control Parameters

Parameter Name	Range	meaning
password_ format_ option	0-31	Indicates password character combination requirements, with a default value of 0, indicating no complexity requirements. The combination can contain one or more of the numbers (1), lowercase characters (2), uppercase characters (4), and other characters (8). 1: The representation must contain numbers. 2: Indicates that it must contain lowercase letters. 4: Indicates that it must contain uppercase letters. 8: The representation must contain other characters. 16: Indicates that it cannot be the same as the username. To limit the combination to the sum of the above values, any combination can be used. Other characters include: # `$& ();| "<.>\~@%^*-_=+[{]}:,?/' The database password is surrounded by single quotes. If the password contains

		single quotes, it needs to be escaped with . For example, limit the inclusion of all types of characters to (1+2+4+8=15). For non English characters, they are classified according to their corresponding ASCII code range.
password_ min_ length	0-65535	Represents the minimum length of the password, with a default value of 0, indicating no length limit.

4.9.2.2 Password Reuse Management

Restrict users from using historical passwords within a specified number of intervals.

The password interval control parameter is a read-only parameter, defined as follows:

Table -466 Password Reuse Control Parameters

Parameter Name	Range	meaning
password_ reuse_ max	0-100	The default value is 0, indicating no control. A positive value of N indicates the allowed password interval, which can only be set if the number of intervals exceeds N.

4.9.2.3 Password validity management

Control the validity period of the password, and after reaching the validity period, the user's password will automatically expire. After the user's password expires, the user is allowed to log in and prompted to change the password when executing SQL. At this point, users are allowed to modify their own passwords. The password must be changed before other SQL can be executed.

The configuration of user password expiration is sent to each coordinator node, and the syntax definition is as follows:

```
CREATE/ALTER USER
    user [auth_option]
    [expiration_option | lock_option | host_option] ...
expiration_ option: {
    PASSWORD EXPIRE
    | PASSWORD EXPIRE DEFAULT
    | PASSWORD EXPIRE NEVER
    | PASSWORD EXPIRE INTERVAL N DAY
}
```

explain

- For non-existent users, the password expiration option can be specified when using create user.
- For existing users, the password expiration option can be specified when using alter user to change the user.
- When a user's password expires, executing SQL will prompt the user to first modify the password, for example:

```
gbase> use test;
ERROR 1840 (HY000): You must reset your password using ALTER USER statement before executing this statement.
```

- The password expiration setting will take effect immediately. If multiple policy options are specified, the last one will take effect.

Optional expiration date control password policy

- Use default password expiration time:

  The default password expiration time is a read-only parameter, defined as follows:

  Table -467 Password Validity Control Parameters

Parameter Name	Range	meaning
password_ life_ time	0-65535	The default value is 0, which means password expiration is disabled. A positive value of N indicates the number of days the password has expired, and the password must be modified after N days.

  Example of using default password expiration policy:

```
CREATE USER jeffrey PASSWORD EXPIRE DEFAULT;// Creating users while meeting the default password expiration time
ALTER USER jeffrey PASSWORD EXPIRE DEFAULT;// Modify users to meet the default password expiration time
```

- Disable password expiration policy:

  At this point, the password will never expire. Example:

```
CREATE USER jeffrey PASSWORD EXPIRE NEVER;
ALTER USER jeffrey PASSWORD EXPIRE NEVER;
```

- Specify password expiration interval:

  Set the interval between password expiration to N days. example:

```
CREATE USER jeffrey PASSWORD EXPIRE INTERVAL 180 DAY;
ALTER USER jeffrey PASSWORD EXPIRE INTERVAL 180 DAY;
```

- Expire password immediately:

Expire password immediately, example:

```
CREATE USER jeffrey PASSWORD EXPIRE;
ALTER USER jeffrey PASSWORD EXPIRE;
```

4.9.3 User Security Management

4.9.3.1 Login retry lock

Login retry lock is used to limit the number of login retries for users. As long as a user logs in and reports an error, the number of retries will accumulate. When the number of retries reaches the specified upper limit, the account will be locked. Users in a locked state will be prompted when logging in that their account is locked and will be denied login.

The user login retry count parameter is a read-only parameter, defined as follows:

Table -468 Login retry count parameters

Parameter Name	Range	meaning
login_ attempt_ max	0-65535	The default value is 0, which means password retry is disabled and login information is not recorded. A positive integer value of N indicates the number of times password errors are allowed, and if the number of times is reached, the user account will be locked.

4.9.3.2 Account locking and unlocking

After the user account is locked, it will not be allowed to log in and the account will not be automatically unlocked. An administrator with CREATE USER permission must unlock the user before logging in. Locking only affects user login.

Send the locked user account to each coordinator node.

Grammar format

```
CREATE/ALTER USER
  user [auth_option]
  [expiration_option | lock_option | host_option] ...
lock_ option: {
    ACCOUNT LOCK
  | ACCOUNT UNLOCK
}
```

Command Description

- For non-existent users, using create user can be set to either locked or non locked state.
- For existing users, use alter user to change the user to a locked or non locked state.
- When a locked account logs in, it prompts the user to lock it and refuses to log in, such as:

```
gbase -uuser1
ERROR 1830 (HY000): Access denied for user 'user1'@'%'. Account is locked.
```

- If multiple policy options are specified, the last one takes effect. Dbauser users are not subject to lock control and can lock and unlock, but are still allowed to log in when locked.

4.9.3.3 Account Restricted Host List

Grammar format

```
CREATE/ALTER USER
user [auth_option]
[expiration_option | lock_option | host_option] ...
host_ option: {
hosts 'host_ list'
}
```

Command Description

host_ The value of the list can be IP or host name, and multiple characters are allowed to be included. Use the space "" to separate it, and use "%" and "_" as wildcards (the wildcard usage is the same as the original host function). Default host_ The list is empty, and there is no host restriction for login. The login IP or host name is only allowed in the list for users to log in.

Specify when creating a user user@host When host is a specific fixed and unique string, you cannot specify the account login list hosts here. If you must specify it, the hosts should be the same as host.

4.9.4 View security information

From gbase.user_ User security information can be queried in the check system table.

user_ The check table structure is as follows:

Table -469 Users_ Check Table Structure

Column Name	meaning
host	Host name, primary key
user	Username, primary key
attempt	Number of password retries
last_ attempt	Number of recent successful login retries
locked	Is the user locked
password_ expired	Does the password expire
password_ last_ changed	Last password modification time
password_ life_ time	Password validity period, in days
password_ history	Password history list, ciphertext
host_ list	List of hosts allowed to log in
login_ time	Current login time
login_ host	This login to the host
last_ login_ time	Last login time
last_ login_ host	Recently logged in to the host
login_ count	Number of user logins

Example

Query the user's lock and password expiration status.

```
gbase> select Locked,password_ expired from gbase.user_ check where user =
'user1';
+--------+------------------+
| Locked | password_ expired |
+--------+------------------+
| N      | N                |
+--------+------------------+
```

4.9.5 Login information display

Login information display is subject to read-only parameter show_ login_ Status control:

Table -470 Parameter Description

Parameter Name	Range	meaning
show_ login_ status	0-1	The default value is 0, indicating shutdown; 1 indicates on.

Example

Example 1: When the login information display is enabled, the following information can be displayed when the user logs in.

```
Login info:
                    USER: root
                LOGIN_ TIME: NULL
```

```
                LOGIN_ HOST:
        LAST_ LOGIN_ TIME: NULL
        LAST_ LOGIN_ HOST:
                LAST_ RETRY: 0
VALID_ PASSWORD_ EXPIRE: NEVER
```

Example 2: Display the user's own login information.

```
gbase> show login status\G
********************** 1. row **********************
                        USER: user1
                LOGIN_ TIME: 2017-10-25 09:38:50
                LOGIN_ HOST: localhost
        LAST_ LOGIN_ TIME: 0000-00-00 00:00:00
        LAST_ LOGIN_ HOST:
                LAST_ ATTEMPT: 0
VALID_ PASSWORD_ EXPIRE: 1
```

4.9.6 Permissions for security management

The security of the GBase8a MPP Cluster database depends on the permission management system. The permission management system conducts security management for all connections, queries, and other operations based on the access control list. The permission management system authenticates users who connect to the GBase8a MPP Cluster database instance and the permissions they have. The permission system ensures that users only perform what is allowed. When a user connects to the server, their identity is jointly determined by the host initiating the connection and the specified username. After successful connection, the command is initiated, and the permission system determines whether to grant execution permission based on the user's identity and the type of command initiated. The execution of the CREATE USER syntax is controlled by the CREATE USER permission. View security information in the system table gbase.user_ SELECT permission control for check.

The execution of the Alter USER syntax is controlled by the CREATE USER permission, except for modifying one's own password. Modifying one's own password is not controlled by permission.

4.9.7 Login Management Consistency

Password security management within a cluster requires consideration of consistency issues and faces the following scenarios.

Assuming that the cluster includes three coordinators, A, B, and C:

Scenario 1: Login at point A, reaching the number of password locks, point A will be locked, while point B and point C will also be locked.

Scenario 2: When logging in at point A and experiencing a password retry error, even if no password retry is performed at point B and point C, the number of password retries will increase.

Scenario 3: Login at point A, password verification successful, password retry count needs to be cleared to 0. If the retry count at point B and point C is not 0, it also needs to be cleared to 0.

Due to scenario 3, clearing the number of password retries at all nodes for every login will greatly affect efficiency. Moreover, during the zero clearing operation, the feventlog will be recorded. If any nodes are offline, the feventlog will increase significantly, which will have a significant impact on the system. Therefore, add a parameter for control, which is turned off by default. After understanding the above effects, it can be manually turned on.

Table -471 Parameter Description

Parameter Name	Range	meaning
gcluster_ user_ check_ consistent	0、1	0 indicates off, 1 indicates on, and defaults to on.

4.9.8 Client Access Authentication

4.9.8.1 Client connects to the cluster using SSL encryption

To protect the security of sensitive data transmission, GBase 8a MPP Cluster supports communication between clients and servers through SSL encryption.

prerequisite

The encryption function requires the installation of the openssl library in the system and the ability to execute openssl commands.

background information

GBase 8a MPP Cluster supports the SSL standard protocol, which is a higher sec urity protocol standard that incorporates digital signatures and certificates to achie ve bidirectional authentication between clients and servers, ensuring more secure d ata transmission for both communication parties.

4.9.8.1.1 Generate SSL connection certificate

Operating Steps

In the system on the server side of the cluster, select the directory to generate SSL keys as needed, taking the path/usr/local/ssl as an example.

Generate server side keys and certificates

Step 1

Execute the following command to enter the directory.

```
$ cd /usr/local/ssl
```

Step 2

To generate a ca cert. pem, you need to fill in information such as Country Name. You can fill it in the following way or according to the user's actual situation.

```
$ openssl req -sha1 -new -x509 -nodes -days 3650 -keyout ca-key.pem >
ca-cert.pem
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:11
State or Province Name (full name) []:1
Locality Name (eg, city) [Default City]:1
Organization Name (eg, company) [Default Company Ltd]:1
Organizational Unit Name (eg, section) []:1
Common Name (eg, your name or your server's hostname) []:1
Email Address []:1
```

Generate a key and also fill in some information. In the password section (A challenge password []:), it is recommended to fill in a more complex password.

```
$ openssl req -sha1 -newkey rsa:2048 -days 730 -nodes -keyout
server-key.pem > server-req.pem
Generating a 2048 bit RSA private key
.........+++
.....+++
```

```
writing new private key to 'server-key.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:11
State or Province Name (full name) []:1
Locality Name (eg, city) [Default City]:1
Organization Name (eg, company) [Default Company Ltd]:1
Organizational Unit Name (eg, section) []:1
Common Name (eg, your name or your server's hostname) []:1
Email Address []:1

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:123456
An optional company name []:1
```

Step 3

Export server-key.pem as an RSA type.

```
$ openssl rsa -in server-key.pem -out server-key.pem
writing RSA key
```

Step 4

Generate server-cert.pem.

```
$ openssl x509 -sha1 -req -in server-req.pem -days 730 -CA ca-cert.pem
-CAkey ca-key.pem -set_ serial 01 > server-cert.pem
Signature ok
subject=/C=11/ST=1/L=1/O=1/OU=1/CN=1/emailAddress=1
Getting CA Private Key
```

Generate client side keys and certificates

Step 1

In the same directory, generate the client side key and certificate, generate the key, and input the same information as the server side.

```
$ openssl req -sha1 -newkey rsa:2048 -days 730 -nodes -keyout
client-key.pem > client-req.pem
Generating a 2048 bit RSA private key
.....+++
.....+++
writing new private key to 'client-key.pem'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
      -----
Country Name (2 letter code) [XX]:11
State or Province Name (full name) []:1
Locality Name (eg, city) [Default City]:1
Organization Name (eg, company) [Default Company Ltd]:1
Organizational Unit Name (eg, section) []:1
Common Name (eg, your name or your server's hostname) []:1
Email Address []:1

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:123456
An optional company name []:1
```

Step 2

Export client key.pem as an RSA type.

```
$ openssl rsa -in client-key.pem -out client-key.pem
writing RSA key
```

Step 3

Generate client-cert.pem.

```
$ openssl x509 -sha1 -req -in client-req.pem -days 730 -CA ca-cert.pem
-CAkey ca-key.pem -set_ serial 01 > client-cert.pem
Signature ok
subject=/C=11/ST=1/L=1/O=1/OU=1/CN=1/emailAddress=1
Getting CA Private Key
```

4.9.8.1.2 Server Configuration

Operating Steps

Step 1

Modifying the cluster configuration file gbase_ 8a_ Gcluster.cnf, add ssl information in [gbased]. Taking the path/usr/local/ssl as an example, add the following example:

```
$ vi $GCLUSTER_ BASE/config/gbase_ 8a_ gcluster.cnf
[client]
port=5258
socket=/tmp/gcluster_ 5258.sock
connect_ timeout=43200
#default-character-set=gbk

[gbased]
basedir = /opt/gcluster/server
datadir = /opt/gcluster/userdata/gcluster
socket=/tmp/gcluster_ 5258.sock
pid-file = /opt/gcluster/log/gcluster/gclusterd.pid

#default-character-set=gbk

ssl-ca=/usr/local/ssl/ca-cert.pem
ssl-cert=/usr/local/ssl/server-cert.pem
ssl-key=/usr/local/ssl/server-key.pem

log-error
port=5258
```

```
core-fil
```

Step 2

Check if the configuration is successful and restart the cluster.

```
#gcluster_ services all restart
```

Step 3

Execute gccli and log in to the cluster.

```
$ gccli -uroot

GBase client 9.5.2.13.113642. Copyright (c) 2004-2020, GBase.    All Rights
Reserved.
```

Step 4

Check the ssl parameter status, and if it is successfully configured, it will be displayed as' YES'.

```
gbase> show variables like 'have_% ssl';
    +---------------+-------+
    | Variable_ name | Value |
    +---------------+-------+
    | have_ openssl    | YES     |
    | have_ ssl            | YES    |
    +---------------+-------+
2 rows in set
```

4.9.8.1.3 Client Configuration

Operating Steps

Step 1

Copy the server generated ca cert. pem, client req. pem, client key. pem, and clie nt cert. pem to the client and modify the cluster configuration file gbase_ 8a_ Gc luster.cnf, add ssl information in [client], using path/usr/local/ssl as an example, as shown in red in the following example:

```
$ vi $GCLUSTER_ BASE /config/gbase_ 8a_ gcluster.cnf
[client]
```

```
port=5258
socket=/tmp/gcluster_ 5258.sock
connect_ timeout=43200
#default-character-set=gbk

ssl-ca=/usr/local/ssl/ca-cert.pem
ssl-cert=/usr/local/ssl/client-cert.pem
ssl-key=/usr/local/ssl/client-key.pem

[gbased]
basedir = /opt/gcluster/server
datadir = /opt/gcluster/userdata/gcluster
socket=/tmp/gcluster_ 5258.sock
pid-file = /opt/gcluster/log/gcluster/gclusterd.pid

#default-character-set=gbk

log-error
port=5258
core-file
```

Step 2

Remote access to the server through the client side. For example, logging in to t he server on 192.168.134.131 using the ssluser user:

```
[ gbase@localhost    config]$ gccli -h192.168.134.131 -ussluser -p
Enter password:
```

Step 3

Run the status command, and the SSL section displays "Cipher in use", indicating that the SSL encryption connection was successful. The specific example is as f ollows:

```
[ gbase@localhost    config]$ gccli -h192.168.134.131 -ussluser -p
Enter password:
```

Step 4

If the client side does not have the above configuration, it will still connect to th e server by default. The following parameters can be used to enforce the use of

SSL.

```
grant all on testdb.* to ssl_ user require ssl
```

4.9.9 User level disk quota

Function:

User level disk quotas limit the sys of tables created by users by setting their disk usage limit_ The sum of tablespace and metadata spaces.

Grammar format:

Grant usage on *.* to <user name> limit_ storage_ size=integer[K,M,G,T]

To view user disk quotas and disk usage status:

select * from information_ schema.gnodes_ user_ diskspace_ usage;

Control refresh system table disk usage status:

- _ gbase_ storage_ flush_ interval

When the database service starts and shuts down, it will automatically refresh the disk usage status record into the corresponding system table. This parameter then controls the number of seconds between refreshing the disk statistics information of the system table. The minimum value of this parameter is 1, and the maximum value is 86400 (24 hours). This parameter can be customized using the following SQL, for example:

gbase> set global _ gbase_ storage_ flush_ interval=100;

- Refresh user storage usage;

Manually refresh the SQL of the disk usage status in the system table.

Example

```
gbase> create user u;
gbase> grant all on *.* to u;
gbase> grant usage on *.* to u limit_ storage_ size=1M;
gbase> select * from information_ schema.gnodes_ user_ diskspace_ usage;
+-----------+-----------------+-------------------------+-------------------+
| NODE_ NAME | User                    | user_ limit_ storage_ size | user_ storage_
size |
+-----------+-----------------+-------------------------+-------------------+
```

```
| node1        | root                  |                                |
96773982 |
| node1        | gbase                 |                                |
0 |
| node1        | u                     | 1M                             |
0 |
| node2        | root                  |                                |
96461010 |
| node2        | gbase                 |                                |
0 |
| node2        | u                     | 1M                             |
0 |
| node3        | root                  |                                |
96548104 |
| node3        | gbase                 |                                |
0 |
| node3        | u                     | 1M                             |
0 |
| node4        | root                  |                                |
96860988 |
| node4        | gbase                 |                                |
0 |
| node4        | u                     | 1M                             |
0 |
+-----------+------------------+-------------------------+-------------------+
12 rows in set (Elapsed: 00:00:00.01)
[ gbase@rhel73-1    ~]$ gccli –uu
gbase> use ssbm_ u;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> create table part (
        p_ partkey      bigint,
        p_ name          varchar(22),
        p_ mfgr          varchar(6)    comment 'lookup',
        p_ category     varchar(7)    comment 'lookup',
        p_ brand1        varchar(9)    comment 'lookup',
        p_ color         varchar(11) comment 'lookup',
        p_ type          varchar(25) comment 'lookup',
        p_ size          int,
        p_ container    varchar(15) comment 'lookup'
        );
Query OK, 0 rows affected (Elapsed: 00:00:01.60)

gbase> load data infile    ' file://192.168.146.20/opt/ssbm/part.tbl ' into table
part data_ format 3    FIELDS TERMINATED BY '|';
```

```
Query OK, 200000 rows affected (Elapsed: 00:00:05.69)
Task 262151 finished, Loaded 200000 records, Skipped 0 records

gbase> load data infile    ' file://192.168.146.20/opt/ssbm/part.tbl ' into table
part data_ format 3    FIELDS TERMINATED BY '|';
ERROR 1733 (HY000): (GBA-01EX-700) Gbase general error: Task 262152
failed, [192.168.146.20:5050](GBA-02AD-0005)Failed to query in gnode:
DETAIL: (GBA-01EX-700) Gbase general error: (gns_host: ::ffff:192.168.146.21)
The disk space of User 'u' has exceeded the limit value. (Usage: 4437691,
Limit: 1048576)
SQL: LOAD /*+ TID('262223') */ DATA INFILE '/opt/ssbm/part.tbl' INTO
TABLE `ssbm_ u`.` part_ n1` DATA_ FORMAT 3 FILE_ FORMAT
UNDEFINED    FIELDS TERMINATED BY '|' HOST '::ffff:192.168.146.20'
CURRENT_ TIMESTAMP 1601029204 SCN_ NUMBER 2
gbase> select * from information_ schema.gnodes_ user_ diskspace_ usage;
+-----------+-----------------+------------------------+-------------------+
| NODE_ NAME | User                    | user_ limit_ storage_ size | user_ storage_
size |
+-----------+-----------------+------------------------+-------------------+
| node1         | root                    |                                    |
96773982 |
| node1         | gbase                   |                                    |
0 |
| node1         | u                        | 1M                                |
4435201 |
| node2         | root                    |                                    |
96461010 |
| node2         | gbase                   |                                    |
0 |
| node2         | u                        | 1M                                |
4437691 |
| node3         | root                    |                                    |
96548104 |
| node3         | gbase                   |                                    |
0 |
| node3         | u                        | 1M                                |
4438453 |
| node4         | root                    |                                    |
96860988 |
| node4         | gbase                   |                                    |
0 |
| node4         | u                        | 1M                                |
4435963 |
+-----------+-----------------+------------------------+-------------------+
```

```
12 rows in set (Elapsed: 00:00:00.17)
```

explain:

- The user level disk quota is user level, and multiple tables created by the user share this disk space.
- User level disk quota only counts the sys of user created express engine tables_ The space occupied by tablespace and metadata is not counted, and the disk usage of tables in the gbase, gclusterdb, and gctempdb libraries is not counted. Library tables of other engines are not counted.
- The disk quota is checked for excess at the beginning of SQL execution, and no error is reported during the execution process if the disk usage exceeds the limit.
- Checking the SQL for disk quota will increase the counted disk size. For simple select, delete, drop, shrink, and other SQL that will not increase the disk size, no excess check will be performed.
- DDL does not check disk quotas.

be careful

User disk quota and resource pool disk management are created from two dimensions for the express engine table sys created by users_ The disk space occupied by tablespace and metadata is limited, and they are independent of each other. When enabled simultaneously, exceeding the minimum limit for both will result in an error.

4.9.10 data encryption

4.9.10.1 Overview of Data Encryption

GBase 8a MPP Cluster data encryption provides soft encryption function for database landing data to meet user security needs and improve system security. Data encryption is carried out based on the minimum unit of DC in the data file, which can achieve encryption requirements of different granularity at the table or column level. All encrypted data in the database uses the same key, and the system will automatically read the content of the created key file upon startup. After the data is encrypted and decrypted

using the key file content, the key file content cannot be changed again.

The operations supported by data encryption are as follows:

- Support multiple encryption algorithms such as national security encryption;
- Support encrypted keyword encryption for table creation;
- Support encryption requirements of different granularity at the table or colum n level;
- Support query of table encryption attributes;
- Support key certificate management, including creation, opening, closing, pass word modification, and key conversion of key certificates

  Operation;
- Support key type conversion, i.e. from plaintext key to ciphertext key, or fro m ciphertext key to plaintext key

  Text key:
  - Clear text key: No user password required, can be randomly generated or manually entered;
  - Cryptographic key: The user must input a password to encrypt and store the randomly generated key based on the password;
- Support querying the current key certificate status;
- Support row and column encryption.

4.9.10.2 Data encryption operation

4.9.10.2.1 Specify encryption algorithm type

GBase 8a MPP Cluster supports AES and the National Security SM4 algorithm (symmetric encryption algorithm) for encryption and decryption during data storage and reading.

By changing parameters in the configuration file_ gbase_ encrypt_ new_ The value of mode is used to specify the encryption algorithm type. Value is 1, AES encryption and decryption; The encryption and decryption of SM4 with a value of 2.


be careful

- After the algorithm has been specified, it is not allowed to replace other security algorithms. If the security algorithm flag is forcibly replaced, the original key cannot be decrypted correctly using the new algorithm.
- If the parameter exceeds 2, the service cannot start.
- The default value is 1, and the parameter can only be modified in the configuration file. Used in the global scope.

4.9.10.2.2 Create encrypted tables and columns

Users can create tables or columns with encryption attributes using the create table and encryption keywords, or create tables using the create table like command. However, the alter command is not supported to add encryption attributes to tables or columns.

Example

Create an encrypted representation, for example:

```
create table t1 (a int, b varchar(5)) encrypt;
```

An example of creating an encrypted column is as follows:

```
create table t1 (a int, b varchar(5) encrypt);
```

4.9.10.2.3 View encryption properties

Use show create table to query encrypted attributes. The encrypted attributes of the table will be passed to the columns, and the encrypted attributes of the columns will not affect the table.

```
gbase> create table tb(a int encrypt);
Query OK, 0 rows affected (Elapsed: 00:00:00.03)
gbase> show create table tb;
+-------+-----------------------------------------------------------------------+
| Table | Create Table                                                          |
+-------+-----------------------------------------------------------------------+
| tb    | CREATE TABLE "tb" (
```

```
  "a" int(11) DEFAULT NULL ENCRYPT
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace' |
+-------+-------------------------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.07)
```

4.9.10.2.4 Key Certificate Management

The management of key certificates requires administrator users to process them through SQL. The certificate is stored in the config directory, and in a cluster environment, both gnode and gcluster generate the same key certificate file:

```
gnode：$GBASE_ BASE/config/encryption.crt

gcluster：$GCLUSTER_ BASE/config/encryption.crt
```

be careful

- It is recommended to backup the key certificate. If the certificate is lost, it will affect the decryption operation of existing data;
- Only after opening the ciphertext key certificate can dml operations be performed on the encrypted column;
- After closing the ciphertext key certificate, all dml operations on encrypted columns will become invalid;
- There is only one key certificate, which can only be created once and cannot be repeated;
- The ciphertext key requires the user to remember the password, and the system does not record the password.

4.9.10.2.4.1 Create Certificate

Function Description

Create plaintext and ciphertext key certificates. If password is empty, create plaintext key certificates without requiring a password; If the password is not empty, create a ciphertext key certificate that requires a password; Only one key certificate cannot be created repeatedly.

grammar

```
CREATE ENCRYPTION CERTIFICATE IDENTIFIED BY  'password'
```

```
[CONTENT    'content_value']
```

Table -472 Parameter Description

Parameter Name	Description
password	Key certificate password.
content_ value	Key content keyword, optional. If this keyword is not specified, the system will automatically generate the key when creating it; If specified, the user needs to manually enter the key without any restrictions on the content, with a maximum support of 128 bytes.

Example

```
----------Example of creating a plaintext key certificate
gbase> create encryption certificate identified by '';
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
----------Certificate duplicate creation
gbase> create encryption certificate identified by '';
ERROR 1835 (HY000): encryption certificate already exists.
----------Example of creating a ciphertext key certificate
gbase> create encryption certificate identified by 'ddd22';
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
----------Certificate duplicate creation
gbase> create encryption certificate identified by 'ddd22';
ERROR 1835 (HY000): encryption certificate already exists.
----------Incorrect password detection (password security check enabled, password format_option, password min_length parameters must be configured)
gbase> create encryption certificate identified by 'ddd~';
ERROR 1802 (HY000): Invalid password format,length should >= 4 and contain 'number'.
```

4.9.10.2.4.2 Opening and closing certificates

grammar

```
ALTER ENCRYPTION CERTIFICATE OPEN IDENTIFIED BY 'password'
```

explain

- To open the ciphertext key certificate based on the password, the user needs to enter the password to decrypt the certificate and obtain the encryption key in order to Store data encryption/decryption.

```
ALTER ENCRYPTION CERTIFICATE CLOSE;
```

- Closing the key certificate will prevent data encryption/decryption, which will affect the DML operation of the encrypted column

Note: The plaintext key cannot be closed. You need to convert the plaintext key to a ciphertext key in order to close it.

An example is as follows:

```
----------Open Key Certificate Example
gbase> alter encryption certificate open identified by '1111';
Query OK, 0 rows affected (Elapsed: 00:00:04.76)
----------Repeat open
gbase> alter encryption certificate open identified by '1111';
ERROR 1829 (HY000): encryption certificate already open.
----------Certificate does not exist
gbase> alter encryption certificate open identified by '1111';
ERROR 1829 (HY000): encryption certificate not exists.
----------Decryption failed
gbase> alter encryption certificate open identified by '2222';
ERROR 1829 (HY000): decrypt failed, please check password.
----------Turn off key certificate
gbase> alter encryption certificate close;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> insert into t1 values(4);
ERROR 1838 (HY000): Encrypt key invalid.
gbase> select * from t1;
ERROR 1838 (HY000): Decrypt key invalid.
```

4.9.10.2.4.3 Display certificate status

grammar

```
SELECT * FROM INFORMATION_ SCHEMA.ENCRYPTION_
CERTIFICATE_ STATUS;
```

explain

Add View ENCRYPTION_ CERTIFICATE_ Status, displays the certificate status, as shown in the following example:

```
gbase> SELECT * FROM INFORMATION_ SCHEMA.ENCRYPTION_
CERTIFICATE_ STATUS;
+------+-----------+----------+-------------+
| HOST | IS_ CREATE | KEY_ TYPE | OPEN_ STATUS |
+------+-----------+----------+-------------+
|      | YES       |        0 | ON          |
+------+-----------+----------+-------------+
1 row in set (Elapsed: 00:00:00.00)
```

4.9.10.2.4.4 Change certificate password

grammar

```
ALTER ENCRYPTION CERTIFICATE IDENTIFIED BY ‘old_ pwd’ TO
‘new_ pwd’
```

explain

- Change the ciphertext key password, old_ pwd、new_ Pwd is not empty;
- To improve password security, the original certificate password can be modified without changing encryption

The key is simply re encrypted with a new password to generate a new certificate.

```
gbase> alter encryption certificate identified by '1111' to '2222';
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
----------Old password decryption failed
gbase> alter encryption certificate identified by '1111' to '2222';
ERROR 1829 (HY000): decrypt failed, please check password.
```

4.9.10.2.4.5 Clear text and ciphertext key conversion

grammar

Convert plaintext key to ciphertext key:

```
ALTER ENCRYPTION CERTIFICATE IDENTIFIED BY ‘’ TO ‘password’
```

Convert ciphertext key to plaintext key:

```
ALTER ENCRYPTION CERTIFICATE IDENTIFIED BY ’ password’
TO ‘’
```

explain

- Convert plaintext key to ciphertext key, password cannot be empty;
- Convert plaintext encryption to ciphertext encryption, that is, encrypt the plaintext key as the ciphertext key certificate through a password, and add

  The key used for encryption remains unchanged.
- Convert ciphertext key to plaintext key, password cannot be empty; The ciphertext key is converted into a plaintext key, which is obtained by

  The password decrypts the ciphertext key and obtains the key as the plaintext key, while the key used for data encryption remains unchanged;

```
----------Plaintext key to ciphertext key
gbase> alter encryption certificate identified by ‘’ to ‘2222’;
Query OK, 0 rows affected
----------Certificate does not exist
gbase> alter encryption certificate identified by ‘’ to ‘2222’;
ERROR 1850 (HY000): this syntax is unsupported with current encrypt type
----------Ciphertext key to plaintext key
gbase> alter encryption certificate identified by 'ddd22' to ‘’;
Query OK, 0 rows affected
----------Certificate does not exist
gbase> alter encryption certificate identified by 'ddd22' to ‘’;
ERROR 1850 (HY000): this syntax is unsupported with current encrypt type
```

4.9.10.3 Cluster configuration for data encryption

- Password management for key certificates. If password detection is enabled, a password must be set_

  format_ option、password_ min_ Length, if not enabled, the password rules a nd length are not

  Make restrictions;
- Add encryption to the configuration file in the config directory of the gnode node_ server_ host,

  encrypt_ server_ The port parameter is used to restart the server of the clust er gnode node in the case of ciphertext encryption

  Proactively obtain key data from the server after the transaction;

 **explain**

- encrypt_ server_ Host: Refers to the host IP of gcluster, which can be multiple, separated by commas;
- encrypt_ server_ Port: The server port pointing to gcluster (default 5258).

- If the entire cluster restarts, for ciphertext mode, it is necessary to manually execute open (open key)

The operator can perform related dml operations on encrypted columns.

4.9.11 Data Desensitization

4.9.11.1 background

summary

Due to the fact that sensitive data is an important part of database security, desensitization of sensitive data is necessary. GBase 8a MPP Cluster provides dynamic data desensitization function for different users to meet different needs.

- This enables developers or database administrator to effectively control the exposure of sensitive data in the database, and generate desensitized data at the database level, greatly simplifying the security design and coding of the business application layer.
- Enable users to add desensitization attributes to fields that require data desensitization through SQL syntax, and determine whether to expose raw data to users with query requirements through user permission control.

4.9.11.2 Desensitization function

summary

Dynamic data desensitization does not truly alter the actual data stored in the table, but rather applies this feature to control the data returned during queries.

Whether dynamic data desensitization is enabled is affected by the current user permissions:

Super users and users with unmask permission can access actual data without being affected by desensitization rules;

Users without unmask permission can only access desensitized data due to the impact of desensitization rules;

The table data contained in the warning information of SQL executed by users without

unmask permission is displayed as' ****** ';

Desensitization is only effective for projection columns.

4.9.11.2.1 Grammar format

```
MASKED With (Function='TYPE ')
```

Dynamic data desensitization supports five types of data desensitization functions, as follows:

1. Default desensitization default type.

   This type has no parameters.

   ```
   MASKED WITH(FUNCTION = 'DEFAULT()')
   ```

2. Random type of desensitization.

   The two parameters min and max of random (min, max) define the range of random values, and min and max are limited by the definition range of the field. Min is less than max, min and max can be floating point numbers.

   ```
   create table t1 (a int masked with(function='random(-2147483647,2147483647)'));
   ```

3. Custom desensitization partial type.

   This type contains three parameters, partial (prefix, padding, suffix), which are detailed as follows:

   - Prefix represents the number of reserved display characters for the prefix;
   - Padding represents desensitized display characters;
   - Suffix represents the number of displayed characters that are retained at the end.

   ```
   MASKED WITH(FUNCTION = 'PARTIAL(1,'XXXX',1)')
   ```

4. Hash desensitization sha type.

   This type has no parameters.

   ```
   MASKED WITH(FUNCTION = 'SHA()')
   ```

5. Specify the position desensitization keymask type.

   keymask(substr,padding,pos)

   ```
   masked with(function='keymask("@gbase","****",0)')
   ```

4.9.11.2.2 Function Introduction

4.9.11.2.2.1 Default Desensitization Function

explain

The default desensitization function desensitizes data columns of basic types.

1. If the data type includes date, datetime, and time.
   - The date will be displayed as "1900 01 01";
   - The date time will be displayed as "1900 01 01 00:00:00";
   - The time will be displayed as' 00:00:00 '.
2. If the data type is integer, floating-point, and decimal.
   - Integer and floating-point types will display 0;
   - Decimal will be displayed as 0.000... with 0 decimal places (defined or evaluated types) in the result.
3. If the data type is a string type.

   Will be replaced with fixed 4 X characters' xxxx '.
4. NULL value.

   Display as NULL without desensitization treatment.
5. SQL function.

   If any parameter of the SQL function contains a desensitization attribute, default desensitization is performed based on the returned result type of the function.

Example

```
gbase> CREATE TABLE t_ m_ default(name VARCHAR(10) MASKED WITH(FUNCTION = 'DEFAULT()'),b_ date DATETIME MASKED WITH(FUNCTION = 'DEFAULT()'),age INT MASKED WITH(FUNCTION = 'DEFAULT()'));
Query OK, 0 rows affected (Elapsed: 00:00:00.53)

gbase> INSERT INTO t_ m_ default VALUES('Jone smith','1989-03-04 12:31:24.123000',29);
Query OK, 1 row affected (Elapsed: 00:00:00.12)

gbase> SELECT * FROM t_ m_ default;
+------+---------------------+------+
| name | b_ date             | age  |
+------+---------------------+------+
| xxxx | 1900-01-01 00:00:00 |    0 |
+------+---------------------+------+
1 row in set (Elapsed: 00:00:00.09)
```

Table -473 Single row data

Name (varchar)	Date of birth (date time)	Age (int)
Jone smith	1989-03-04 12:31:24.123000	twenty-nine

Table -474 Data Display after Application Default Desensitization

Name (varchar)	Date of birth (date time)	Age (int)
xxxx	1900-01-01 00:00:00	0

4.9.11.2.2.2Random desensitization

explain

The random desensitization function only works on numeric types.

It will randomly display numbers as values within a specified range, and if executed multiple times, the random values on the same row will be different.

Example

Assuming the desensitization range is set to (1, 4) first.

```
gbase> CREATE TABLE t_ m_ random(age INT MASKED WITH(FUNCTION = 'RANDOM(1,4)'));
Query OK, 0 rows affected (Elapsed: 00:00:00.09)

gbase> INSERT INTO t_ m_ random VALUES(29),(19);
Query OK, 2 rows affected (Elapsed: 00:00:00.07)
Records: 2  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t_ m_ random;
+------+
| age  |
+------+
|    4 |
|    3 |
+------+
2 rows in set (Elapsed: 00:00:00.03)
```

Table -475 Data before Desensitization

Age (int)
twenty-nine
nineteen

Table -476 Results after applying random desensitization

Age (int)
four
three

be careful

The NULL value is not desensitized and remains displayed as NULL.

4.9.11.2.2.3Custom desensitization

explain

User defined desensitization is to desensitize the character column. The user can set three parameters: the number of characters reserved at the beginning of prefix, the number of characters reserved at the end of suffix, and the masking characters of padding. If the actual content length is less than or equal to prefix+suffix+length (padding), the character content of padding will be displayed directly.

Example

Set the prefix to 3, the suffix to 6, and the padding character "XXXX".

```
gbase> CREATE TABLE t_ m_ partial(context VARCHAR(255) MASKED WITH(FUNCTION = 'PARTIAL(3,"XXXX",6)'));
Query OK, 0 rows affected (Elapsed: 00:00:00.24)

gbase> INSERT INTO t_ m_ partial VALUES('This is a book on the desk.'),('Hello');
Query OK, 2 rows affected (Elapsed: 00:00:00.08)
Records: 2  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t_ m_ partial;
+---------------+
| context       |
+---------------+
| ThiXXXX desk. |
| XXXX          |
+---------------+
2 rows in set (Elapsed: 00:00:00.02)
```

Table -477 Data before Desensitization

Content (varchar (255))
This is a book on the desk.
Hello

Table -478 Results after applying custom desensitization

Content (varchar (255))
ThiXXXX desk.
XXXX

be careful

The NULL value is not desensitized and is displayed as NULL.

4.9.11.2.2.4SHA desensitization

explain

SHA desensitization works on character columns and applies the SHA algorithm to the column content.

Example

```
gbase> CREATE TABLE t_ m_ sha(context VARCHAR(255) MASKED WITH(FUNCTION = 'SHA()'));
Query OK, 0 rows affected (Elapsed: 00:00:00.08)

gbase> INSERT INTO t_ m_ sha VALUES('abc');
Query OK, 1 row affected (Elapsed: 00:00:00.10)

gbase> SELECT * FROM t_ m_ sha;
+------------------------------------------+
| context                                  |
+------------------------------------------+
| a9993e364706816aba3e25717850c26c9cd0d89d |
+------------------------------------------+
1 row in set (Elapsed: 00:00:00.03)
```

Table -479 Data before Desensitization

Content (varchar (255))
abc

Table -480 Display Results after Application of SHA Desensitization

Content (varchar (255))
“a9993e364706816aba3e25717850c26c9cd0d89d”

be careful

The NULL value is not desensitized and still displayed as NULL.

4.9.11.2.2.5Keymask desensitization

explain

Specify the character position desensitization function keymask (substr, padding, pos)

Function: Specify the character as the initial counting position, and desensitize within the specified number of digits. The keymask desensitization function is only used for varchar/char columns for desensitization, and other types of columns will use this function to report an error and return.

The parameter description is as follows:

parameter	Parameter Type	Parameter Description
substr	varchar/char	The substring to search for, such as: xiaoming@gbase.cn The substr of is set to '@'. Note that if the substr length is longer than the length of the searched string, an error will be reported.
padding	varchar/char	Represents a string used to overwrite before or after finding the substr position. For example, "xxx", "* * *", etc
pos	int	0/1 coverage direction, 0 represents forward coverage, and 1 represents backward coverage

The desensitization rules are as follows:

1) If no substr is found in the content, no desensitization operation will be performed. And see the reason for the corresponding non desensitization operation in show warnings. For example, substr 'xxxx' is not existing in string 'xxxxxx'.

2) If there are multiple substrs in the string to be searched, only the position of the first occurrence of the substr is processed.

3) If the desensitized string exceeds the column width defined by the field, it is truncated before or after the value of pos.

Example

```
create table t(a varchar(255) masked with(function='keymask("@gbase","****",0)'));
```

Content before desensitization	Content after desensitization
gbase@gbase.cn	****@gbase.cn

Example:

```
create table t(a varchar(14) masked with(function='keymask("@","********",0)'));
```

Content before desensitization (varchar (14))	Content after desensitization (varchar (14))	Truncated content after desensitization (varchar (14))
gbase@gbase.cn	*******@gbase.cn	*****@gbase.cn

```
create table t(a varchar(14) masked with(function='keymask("@","**********",1)'));
```

Content before desensitization (varchar (14))	Content after desensitization (varchar (14))	Truncated content after desensitization (varchar (14))
gbase@gbase.cn	gbase@***************	gbase@********

4.9.11.2.3 Related attributes

You can add or modify the desensitization properties of fields when creating or modifying tables.

1. To create a table, only create permission is required:

```
create table t1(a int masked with(function= 'default()'));
create table t2(a varchar(255) masked with(function= 'sha()'));
create table t3(a int masked with(function= 'random(1,10) '));
create table t4(a varchar(255) masked with(function= 'partial(2,"XXX", 3) '));
```

explain

Create table like inherits mask attributes.

2. Alter can add and modify mask attributes. The syntax and content of the mask are in the same table, and both alter and unmask permissions are required.

```
Alter table t1 alter a masked with(function= 'sha()');
```

3. Display mask properties.

```
Show create table <tablename>;
```

For example:

```
gbase> show create table t4;
+-------+-------------------------------------------------------------
-----------------------------------------------------------------
----------------------------+
| Table | Create Table
|
+-------+-------------------------------------------------------------
-----------------------------------------------------------------
-------------------------+
| t4       | CREATE TABLE "t4" (
   "a" varchar(255) DEFAULT NULL MASKED WITH(FUNCTION='DEFAULT()')
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_ tablespace' |
```

4. Delete the mask attribute.

```
Alter table table name Alter column name drop masked;
```

5. Super users or other users with unmask privileges can view raw undensitized data.

```
grant unmask on *.* to user;
```

For example:

```
create user user2;
grant unmask on *.* to user2;
show grants for user2;
```

result:

```
+-----------------------------------------------+
| Grants for user2@%                            |
+-----------------------------------------------+
| GRANT UNMASK ON *.* TO 'user2'@'%' |
+-----------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

4.9.11.2.4 Explanation of Desensitization of Projection Column Function

4.9.11.2.4.1 Desensitization rules

Functions can be roughly divided into two categories based on different processing logic and different ways of obtaining desensitization rules.

1. The control flow function can directly return the desensitized column, rather than comparing and calculating the desensitized column. In this case, the function will apply the desensitization rules of the desensitized column to other return values. These class function include case when/code, if, ifnull, nvl, and coalesce functions. For example:

```
select case col when 1 then '123' when 2 then '456' when 3 then mask_ col else '789' end from t;
```

explain

Where col is a non desensitized column, mask_ Col is the desensitization column, and "123", "456", and "789" are constants.

The possible return values of the case when function in the above query are '123', '456', '789', and mask_ Col, if the return value contains a desensitization column, the constants' 123 ',' 456 ', and' 789 'will be masked according to the desensitization column_ The desensitization rules of col are used for desensitization.

- When there are multiple desensitized columns (different columns) in the return value of this class function, the function will use the default desensitization rules (regardless of whether the desensitization rules of multiple columns are identical).

For example:

```
select nvl(mask_col1, mask_col2) from t;
```

The nvl function will use the default desensitization rules.

and

```
select nvl(mask_col1, mask_col1) from t;
```

The nvl function will still use mask_ The desensitization rules for col1.

- When the return value of this class function contains both desensitized and non desensitized columns (not constants), the function will also use the default desensitization rules.

  For example:

  ```
  select coalesce(col, mask_col) from t;
  ```

  The coalesce function will use the default desensitization rules.

- When the types in the return values of these class function are not identical, they will be converted according to the conversion rules of data types. If the converted data type is inconsistent with the original type of the desensitized column (only the string, real, decimal, and int levels are considered, and the finer data types are not considered), the default desensitization will be used.

  For example:

  ```
  select coalesce(mask_int_col, 'abcdef') from t;
  ```

  The coalesce function returns a string, which is the same as the mask_ int_ The type (int) of col is inconsistent, so coalesce will use default desensitization.

  When desensitizing by default, the mask is not_ int_ The value of col is converted to a string and instead directly returns the default desensitization value (xxxx) of type string.

  Control flow function type conversion rules:

  - When any parameter is a string, the operation is performed according to the string;
  - When there is no string, if there is a real, perform the operation according to real;
  - When there are no strings and real, if there is a decimal, perform the operation using decimal;
  - The date time type is counted as a string type.

  The control flow function only desensitizes all possible return values and does not desensitize other parameters in the control logic. For example, in the case when function, only then and else are desensitized, and parameters in the case and when branches are not desensitized.

- When performing Union, Intersect, and Minus operations, the selection of the corresponding column's desensitization rules is consistent with the desensitization rules of the control flow function.

For example:

select mask_ int_ col from t union select mask_ int_ col as col from t;

Will use mask_ int_ The desensitization rules of col, while:

select mask_ int_ col from t union select mask_ int_ col+1 from t;

Default desensitization will be used.

2. Most functions compare or calculate parameters, such as some string functions, comparison functions, numeric functions, time and date functions, OLAP functions, and aggregate functions. If the parameters of these class function have desensitized columns, the functions will desensitize the return values using the default desensitization rules, rather than participating in operations after desensitizing the desensitized columns.

   For example:

```
select concat(mask_col, '123') from t;
```

   Among them, mask_ Col is the desensitization column, and '123' is a constant.

   The concat function in the above query does not convert the desensitized mask_ The col string concatenates the constant '123', but instead desensitizes the return value of concat by default. Since the concat return value is of character type, the above results are divided by mask_ Except for the case where col is NULL (where the return value is NULL), always return 'xxxx'.

   For example, select mask_ col > 1 from t; The default desensitization rule will be used based on the return value type of function ">", so except for mask_ Col always returns 0 unless it is a NULL value.

   If the maximum length of the desensitized column is exceeded after desensitization, it will be automatically truncated to the maximum length of the desensitized column.

   For example, the desensitization column is defined as:

   mask_ col varchar(5) masked with (function ='partial(2,"xxxx",2)')

   The value "abcde" should theoretically be desensitized to "abxxxxde". If the length after desensitization exceeds the maximum length of 5, it will automatically be truncated to "abxxx".

   Similarly, for some functions with adjustable length, they will also be truncated during desensitization, such as:

   select left(mask_col, 2) from t;

   The left function will use the default desensitization, which should theoretically be 'xxxx', but it exceeds the maximum length set by the left function, so it will be automatically truncated to 'xx'.

4.9.11.2.4.2Inheritance of desensitization rules

1. When nested functions are used, the desensitization rules of the inner function can

be passed to the outer layer.

This transfer is also limited to functions of non comparative computational classes.

For example:

```
select case col when 1 then nvl(mask_col, '123') when 2 then '456' else '789' end from t;
```

explain

Among them, col is a non desensitized column. mask_ Col is the desensitization column, and "123", "456", and "789" are constants.

In the above query, the possible return values of the nvl function include mask_ Col and '123', so the nvl function uses mask_ The desensitization rule of col on mask_ Col and '123' are desensitized, and this desensitization rule is also passed to the outer case when function, so '456' and '789' will also use masks_ The desensitization rules of col desensitize.

2. Inheritance of desensitization rules also occurs in the usage scenarios of subqueries, where the desensitization rules of the outer projection column are inherited from the desensitization rules of the subquery projection column.

   For example:

```
select col from (select reverse(mask_col) as col from t3) as tmp;
```

   Among them, mask_ Col is the desensitization column.

   In the above query, the reverse function will compare or operate on parameters. When the parameter is a desensitized column, the desensitization rule of the function is the default desensitization. The outer layer's col desensitization rule inherits from the desensitization rule of the subquery reverse (mask_col), so the outer layer's col desensitization rule is also the default desensitization rule.

4.9.11.2.4.3Scenario Example

To facilitate the understanding of the above concepts, the following table provides examples of queries and result sets for each typical scenario. The table building statements and data used in the use case are:

```
create table t (i1 int,
vc1 varchar(10) masked with (function='partial(2,"*****",0)'),
vc2 varchar(10) masked with (function='partial(2,"*****",0)'));
insert into t values (1, 'nblknabpa', 'pombkaia');
insert into t values (2, '.mapkna', '0jbadflk');
insert into t values ();
```

- Example 1: The case when desensitization column rule of the ith class function function;

```
gbase>select case i1 when 1 then vc1 when 2 then '12345' else '67890' end as res from t;
+---------+
| res     |
+---------+
| nb***** |
| 12***** |
| 67***** |
+---------+
```

- Example 2: Coalesce desensitization column rule of the i class function;

```
gbase>select coalesce(vc1,'12345') as res from t;
+---------+
| res     |
+---------+
| nb***** |
| .m***** |
| 12***** |
+---------+
```

- Example 3: case when multiple desensitized columns of the ith class function function are desensitized by default;

```
gbase> select case i1 when 1 then vc1 when 2 then vc2 else '67890' end as res from t;
+------+
| res  |
+------+
| xxxx |
| xxxx |
| xxxx |
+------+
```

- Example 4: Class ii class function substring is desensitized by default;

```
gbase> select substring(vc1, 1, 2) as res from t;
+------+
| res  |
+------+
| xx   |
| xx   |
| NULL |
+------+
```

- Example 5: Class ii class function concat is desensitized by default;

```
gbase> select concat(vc1,'123') as res from t;
+------+
| res  |
+------+
| xxxx |
| xxxx |
| NULL |
+------+
```

- Example 6: functions of class i are nested;

```
gbase> select case when i1 > 1 then coalesce(vc1, '12345') else '67890' end as res from t;
+---------+
| res     |
+---------+
| 67***** |
| .m***** |
| 67***** |
+---------+
```

- Example 7: Class ii class function are nested;

```
gbase> select concat(substring(vc1,1,2),'123') as res from t;
+------+
| res  |
+------+
| xxxx |
| xxxx |
| NULL |
+------+
```

- Example 8: Mixed nested call of class i and class ii class function;

```
gbase> select coalesce(substring(vc1,1,2),'12345') as res from t;
(Or select substring (coalesce (vc1,'12345 '), 1, 2) as res from t;)
+------+
| res  |
+------+
| xxxx |
| xxxx |
| xxxx |
+------+
```

- Example 9: Calling class i inside or outside a subquery;

```
gbase> select res from (select coalesce(vc1,'12345')as res from t) as tmp;
(Alternatively, select coalesce (vc1,'12345 ') as res from (select vc1 from t) as tmp;)
+---------+
| res     |
+---------+
| nb***** |
| .m***** |
| 12***** |
+---------+
```

- Example 10: Calling Class II within or outside a subquery;

```
gbase> select res from (select concat(vc1,'123') as res from t) as tmp;
(Alternatively, select concat (vc1,'123 ') as res from (select vc1 from t) as tmp;)
+------+
| res  |
+------+
| xxxx |
| xxxx |
| NULL |
+------+
```

- Example 11: Calling class i inside and outside a subquery;

```
gbase> select case when i1 > 1 then res1 else '67890' end as res from (select i1,coalesce(vc1,
'12345') as res1 from t) as tmp;
+---------+
| res     |
+---------+
| 67***** |
| .m***** |
| 67***** |
+---------+
```

- Example 12: Calling Class II inside and outside a subquery;

```
gbase> select concat(res1,'123') as res from (select substring(vc1,1,2) as res1 from t) as tmp;
+------+
| res  |
+------+
| xxxx |
| xxxx |
| NULL |
+------+
```

- Example 13: Mixed invocation of class i and class ii in sub queries.

```
gbase> select substring(res1,1,4) as res from (select coalesce(vc1, '12345') as res1 from t) as tmp;
(or
gbase> select coalesce(res1, '12345') as res from (select substring(vc1,1,2) as res1 from t) as tmp; )
+------+
| res  |
+------+
| xxxx |
| xxxx |
| xxxx |
+------+
```

explain

If the maximum length of the desensitized column is exceeded after desensitization, it will be automatically truncated to the maximum length of the desensitized column.

For example, the desensitization column is defined as:

```
mask_ col varchar(5) masked with (function ='partial(2,"xxxx",2)')
```

The value "abcde" should theoretically be desensitized to "abxxxxde". If the length after desensitization exceeds the maximum length of 5, it will automatically be truncated to "abxxx".

The desensitization column will also be truncated when desensitizing functions with adjustable lengths, such as:

```
select left(mask_col, 2) from t;
```

The left function will use the default desensitization, which should theoretically be 'xxxx', but it exceeds the maximum length set by the left function, so it will be automatically truncated to 'xx'.

When performing Union, Intersect, and Minus operations, the desensitization rules of the corresponding columns are consistent with the desensitization rules of the control flow function.

For example:

```
select mask_ int_ col from t union select mask_ int_ col as col from t;
```

Will use mask_ int_ The desensitization rules of col, while:

```
select mask_ int_ col from t union select mask_ int_ col+1 from t;
```

Default desensitization will be used.

4.9.12 Kerberos security authentication

4.9.12.1 GBase 8a MPP Cluster Installation of Kerberos Client

The installation and configuration of the Kerberos client are mainly divided into the following steps:

- Install the kerberos client installation package on all cluster nodes of GBase 8a MPP Cluster, and then copy the kerberos client configuration file/etc/krb5.conf from the KDC server to the/etc directory on all cluster nodes;
- GBase 8a MPP Cluster cluster loads or exports HDFS files through kerberos authentication, and the minimum version of the kerberos installation package is required to be 1.10; If it is detected that the current Kerberos version is lower than 1.10, the message "Kerberos version too old, require 1.10 or higher" will be printed in the express.log log;
- Copy the Kerberos authentication key file keytab to the specified directory of all nodes, with the Coordinated node directory being $GCLUSTER_ BASE/config, the data node directory is $GBASE_ BASE/config;
- Append the HTTPS CA root certificate file to the root certificate files of all nodes, where the CA root certificate file contains one or more certificates.
  The root certificate file of the Coordinate node is: $GCLUSTER_ The root certificate file of the BASE/config/ca bundle. crtdata node is: $GBASE_ BASE/config/ca-bundle.crt

4.9.12.2 HDFS file operation under Kerberos security authentication

4.9.12.2.1 Loading/exporting HDFS files under Kerberos authentication

In the case of integrating kerberos security authentication in HDFS, GBase 8a MPP Cluster nodes can load or export HDFS files under kerberos authentication after deploying the kerberos client. The following configurations need to be completed for loading or exporting operations:

- Set gbase_ hdfs_ auth_ Mode=kerberos, specifying the use of kerberos authentication to connect to HDFS.
- Set gbase_ hdfs_ Protocol=http/https/rpc, specifying the use of HTTP/HTTPS/RPC protocol to connect to HDFS.
- Set gbase_ hdfs_ Principal="xxx", specifies the Kerberos authentication principal.
- Set gbase_ hdfs_ Keytab='xxx ', specify the keytab file path.

After the above configuration is completed, the loading and exporting operation can be carried out. The specific operation is the same as the ordinary HDFS file operation,

which can be referred to in sections 5.2.1 and 5.2.2 of this chapter.

Attention should be paid to the configuration before loading and exporting:

- The default HTTP port number for HDFS is 50070, HTTPS port number is 50470, and RPC port number is 9000. The ports for the three protocols are different, and the ports in the URL for loading or exporting SQL need to be consistent with the specified protocol.
- When connecting to HDFS using the HTTPS protocol, because the client needs to use a CA root certificate to verify the HTTPS address, the host name (or address) of the specified HDFS NameNode in the URL for loading or exporting SQL must be exactly the same as the host name (or address) signed by the CA.
- When gbase is not specified_ hdfs_ When the keytab parameter value or the specified parameter value is an empty string, gbase will be used_ hdfs_ Principal infers the keytab file name. At this point, the keytab file should be copied to the config directory, and the name of the keytab file should match the gbase_ hdfs_ Please refer to the configuration file section for the corresponding values of the principal parameter. For example:
  set gbase_ hdfs_ principal='gbase/ namenode@HADOOP.COM The keytab file name in the config directory should be: gbase_ namenode.kt。
- Due to the high dependency of Hadoop and Kerberos on DNS resolution, DNS needs to support forward and reverse lookups. In the kerberos authentication environment, it is recommended to use host names in URLs for loading and exporting statements, rather than IP addresses.
- Configuration for importing and exporting multiple sets of Hadoop clusters with different kerberos authentication
  - Merge multiple sets of kerberos configuration files (merge the krb5 files of multiple kerberos servers into the/etc directory);
  - If there are multiple other files related to Kerberos, place them all in the corresponding directory. For example, multiple keytabs are placed in the corresponding config directory, and multiple CA certificates are also placed in the corresponding config directory;
  - The relevant parameters for importing and exporting Hadoop to GBase 8a cluster are currently only available in GBase_ hdfs_ Namenodes support writing multiple sets of Hadoop clusters, while other parameters only support one set of Hadoop clusters. Therefore, GBase's Hadoop parameters can be dynamically configured through session level parameters or written in GBase using URL parameters_ hdfs_ In namenodes;
  - Since GBase 8a accesses kdc through an API, there is no need to initialize the

kerberos client using kinit.

4.9.12.3 The impact of GBase 8a MPP Cluster installation on Kerberos authentication

- Cluster expansion impact

After the cluster version that supports kerberos authentication is expanded, the administrator needs to perform the installation and configuration of the kerberos client and manually complete the deployment of the kerberos client environment.

- Cluster upgrade impact

To upgrade from a cluster version that does not support Kerberos authentication to a cluster version that does, the administrator needs to perform the installation and configuration of the kerberos client and manually complete the deployment of the kerberos client environment.

- Cluster node replacement tool

  Node replacement function for cluster versions that support kerberos authentication. When the cluster node replacement tool synchronizes files, it needs to add $GCLUSTER_ BASE/config or $GBASE_ Synchronize files with extensions of. kt and. pem/. crt/. cer/. crl under BASE/config to the replaced node.

4.9.12.4 GBase 8a and Kafka Data Source's Kerberos Authentication

When Gnode reads Kafka data and gcluster obtains Kafka's topic metadata information, it must establish a connection as a client with the broker of the Kafka cluster. After version 0.9, the Kafka cluster supports three authentication mechanisms: SSL, SASL/Kerberos, and SASL/PLAIN. GBase 8a adopts SASL/GSSAPI (Kerberos) as the permission system foundation of the cluster, and performs authentication between the broker of the kafka cluster and the 8a cluster (client). For Kafka clusters with kerberos authentication, the following are currently supported:

- Load Kafka data source with kerberos authentication

  Please refer to Chapter 5.2.2.3.9 for loading syntax and examples.

- GBase Consumer supports Kafka data sources with kerberos authentication

  For specific usage methods of GBase kafka consumer, please refer to Chapter 5.2.5.

be careful

Currently, only one set of Kafka clusters with kerberos authentication function is supported

2. GBase cannot use both Kafka cluster and HDFS kerberos authentication with kerberos authentication at the same time

3. The implementation of GBase and Kafka's kerberos authentication requires that the cluster node has already installed the kerberos client. If not installed, please refer to section 4.9.12.1.

GBase 8a and Kafka's kerberos authentication configuration method:

If the Kafka server is configured with kerberos authentication, GBase 8a needs to set the principal and keytab parameters correctly. When GBase 8a connects to the Kafka cluster, the configured principal and keytab will be used for kerberos authentication.

- gbase_ kafka_ principal

  Used to specify the authentication principal name in kerberos. The qualified principal format in the Kafka cluster environment is username/ hostname@REALM.COM Where username must be kafka. This parameter can be set through the 8a configuration file or through set SQL.

- gbase_ kafka_ keytab

  The key table file corresponding to the specified principal, which contains the key information of the authentication subject. This parameter can be set through the 8a configuration file or through the set global method.

4.9.13 Security authentication of 8a and Kafka clusters

There are two ways for 8a to support Kafka cluster encryption:

- Kerberos authentication

For detailed operations, please refer to the section on Kerberos authentication for GBase 8a and Kafka data sources in 4.9.12.4.

- Simple password authentication

After version 0.10 of Kafka, the Simple User Name/Password (SASL/PLAIN) authentication mechanism was introduced, and the 8a supports Kafka's simple user name/password authentication mechanism to authenticate between the 8a cluster and the broker. 8a added configuration parameter gbase_ kafka_ Username and gbase_ kafka_ Password requires configuring these two parameters in the gnode and gcluster configuration files to specify the

username and password used for Kafka authentication. 8a will use the configured username/password for authentication when connecting to Kafka.

Example 1:

Configuration parameter gbase_ kafka_ Username and gbase_ kafka_ Password, execute loading

set global gbase_ kafka_ username='gbase';

set global gbase_ kafka_ password='gbase';

LOAD DATA INFILE ' kafka://192.168.6.95:9092/yuehaoyu?duration=0 ' INTO TABLE test.t1 fields TERMINATED BY '|' DATA_ FORMAT 3;

Successfully loaded and executed

If the parameters are not configured, an error will be reported:

ERROR 1733 (HY000): (GBA-01EX-700) GBase general error: Task 594750 failed, Failed to acquire metadata: Local: Timed out

Example 2:

Loader consumer for loading

Configure gbase_ kafka_ Username and gbase_ kafka_ password

create kafka consumer kafka_ load_ test1 loader topic yuehaoyu brokers '192.168.6.95:9092' duration 3000 into table test.t1 fields TERMINATED BY '|' DATA_ FORMAT 3 null_ value '\N';

start kafka consumer kafka_ load_ test1;

Successfully loaded

If the parameters are not configured, an error will be reported:

ERROR 1707 (HY000): gcluster command error: can not connect to kafka with the brokers and topic you specified

4.10resource management

background information

In the absence of resource management, multiple users and tasks are executed simultaneously, and resource consumption cannot be effectively adjusted, resulting in intense resource competition and excessive consumption among tasks. In such a scenario, the execution of SQL will become slow and unpredictable, and the system will crash due to excessive resource utilization. Therefore, it is necessary to manage and schedule the use of resources, so that SQL tasks can run efficiently and quickly, and the system can be more stable.

4.10.1 Overview of Resource Management Functions

- The GBase 8a MPP Cluster resource management function can reasonably control the CPU, memory, I/O, and disk space resources used by controlled SQL such as SELECT and DML during the operation process, in order to achieve the requirements of reasonable resource utilization and stable system operation. Among them:
  1. CPU: Implement CPU priority and percentage control over controlled SQL, as well as management of SQL concurrency and parallelism.
  2. Memory: Implement control over the upper limit of memory usage for the operator buffer (large heap) used by controlled SQL.
  3. I/O：实现对受控 SQL 使用的 direct I/O 磁盘读写速率上限的控制。
  4. Disk space: Control the amount of disk space occupied by table data files.

be careful

Disk space control only controls the size of disk space occupied by data files, and index files are not within the control range.

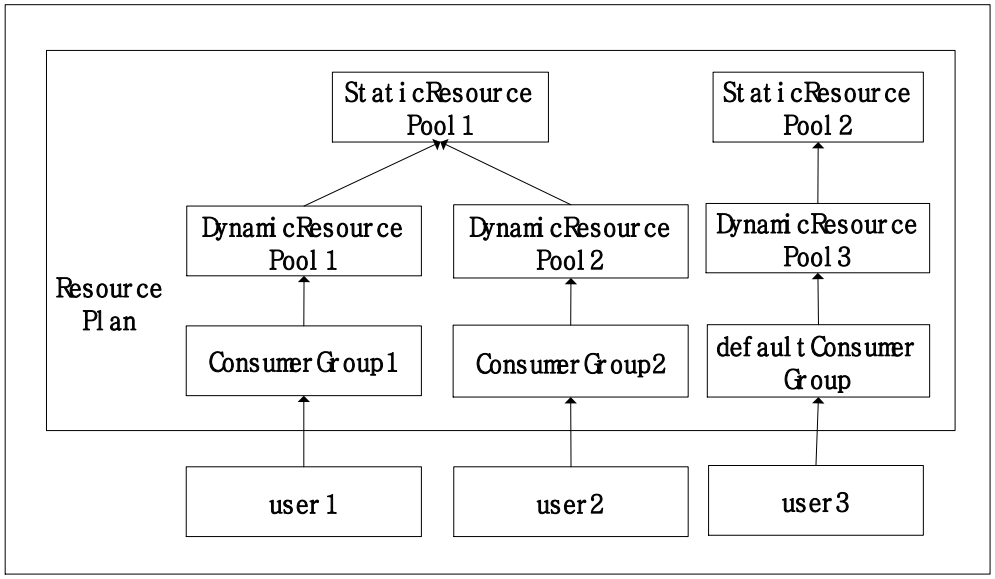
- The resource management functions between each VC are independent of each other, and the resource pool definition, consumption group planning, resource plan status (start/close), and association relationship between consumption groups and resource pools within the VC are all different. Users of different VCs need to configure them according to their own needs.

4.10.2 Resource Management Diagram

GBase 8a MPP Cluster resource management consists of Consumer Group, Resource Pool, Resource Plan, Resource Directive, and User.

The resource management relationship diagram of GBase 8a MPP Cluster is as follows:

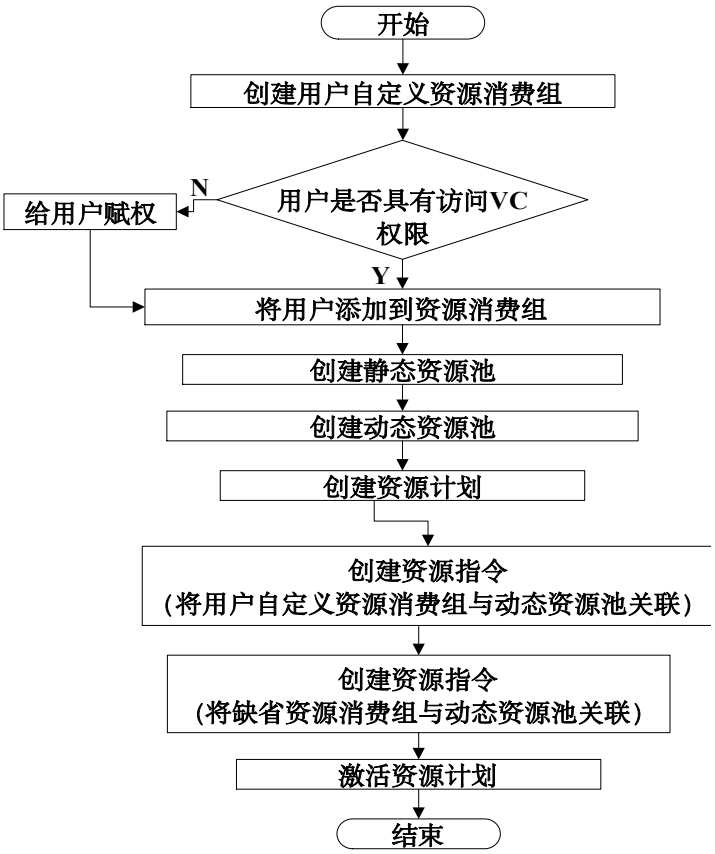
Figure -45 Resource Management Relationship Diagram



- Consumer Group: A collection of multiple users, with a one-to-many relationship between consumer groups and users. Default Consumer Group: A resource consumption group automatically created by the database. Users who have not explicitly joined the custom resource consumption group belong to this group. The Default Consumer Group must manually attach a dynamic resource pool.
- Resource Pool: A collection of several system resources, divided into static resource pool and dynamic resource pool. A static resource pool can contain multiple dynamic resource pools, and a dynamic resource pool can only belong to one static resource pool.
- Resource Plan: A collection that describes the association between consumer groups and resource pools. Under a resource plan, resource pool and consumer group are in a one-to-many relationship, and a resource pool can be associated with multiple consumer groups.
- Resource Directive: describes the association between consumer groups and resource pools in a specified resource plan. There must be an association between the default consumer group and resource pool.

4.10.3 Resource Management Flowchart

Figure -46 Resource Management Flowchart



4.10.4 Resource Management Environment Configuration

Environmental requirements

Table -481 Environmental Requirements

name	Hardware (CPU/RAM/HD)	Operating system and its version	Other software environments
Resource management function	Requirements for hardware operating environment in the same cluster	RedHat 7.3 or later	Install libcgroup and libcgroup level
	Requirements for hardware operating environment in the same cluster	Suse 11 SP3 or later	install Generate and install libcgroup and libcgroup devel after libcgroup1 *. src.rpm

Configuration requirements

- All nodes in the cluster must install the libcgroup and libcgroup level rpm packages. When installing GBase 8a MPP Cluster, it will check whether the rpm package is

installed. If it is not installed, the user needs to manually install the rpm package, otherwise the resource management function cannot be used.

- Ensure that the cgconfig service is started before starting resource management.
- It must be executed before configuring environment variables/ SetSysEnv.py --cgroup

Operating Steps

步骤 1 Install the software package.

```
# rpm –ivh libcgroup-0.37-3.el6.x86_ 64.rpm
# rpm –ivh libcgroup-devel-0.37-3.el6.x86_ 64.rpm
```

步骤 2 Start the service.

In RHEL6:

```
# service cgconfig start
```

In RHEL7:

```
# systemctl start cgconfig.service
```

步骤 3 Check if the cgroup service started successfully.

In RHEL6:

```
# service cgconfig status
```

In RHEL7:

```
# systemctl status cgconfig.service
```

4.10.5 Creating and Managing Consumer Groups

The Consumer Group is called a resource consumption group, and users can divide cluster users into different category groups based on specific business resource usage, with consistent resource consumption behavior among members within the group.

be careful

Virtual cluster version resource management requires granting VC access permissions to users before adding them to the consumption group of resource management.

4.10.5.1 establish

grammar

```
CREATE CONSUMER GROUP [vc_name.]<group_ name> [COMMENT = 'comment'];
```

Table -482 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional parameter. If left blank, the current default VC will be used.
group_ name	Customer group name.
comment	notes.

explain

- Groups in different VCs_ Name allows duplicate names.
- After the cluster installation is completed, there is a default resource consumption group default_ consumer_ Group, the id=1 of the consumption group.
- Any planned default_ consumer_ The group must be associated with a dynamic resource pool.

Example

```
CREATE CONSUMER GROUP vc1.group1 COMMENT = 'test group1';
```

4.10.5.2 change

grammar

- change one's name

```
ALTER CONSUMER GROUP [vc_name.]<group_ name> RENAME [TO] <new_ name>;
```

- Change Description

```
ALTER CONSUMER GROUP [vc_name.]<group_ name> COMMENT = 'comment';
```

- Add members

```
ALTER CONSUMER GROUP [vc_name.]<group_ name> ADD USER <user_ name>;
```

- Delete Members

```
ALTER CONSUMER GROUP [vc_name.]<group_ name> REMOVE USER <user_ name>;
```

Example

- Create User

```
CREATE USER user1;
CREATE USER user2;
```

- New users in resource consumption group

```
ALTER CONSUMER GROUP group1 ADD USER user1;
ALTER CONSUMER GROUP group1 ADD USER user2;
```

- Resource consumption group deleting users

```
ALTER CONSUMER GROUP group1 REMOVE USER user1;
ALTER CONSUMER GROUP group1 REMOVE USER user2;
```

 **be careful**

Under the same VC, there is a one-to-many relationship between consumer groups and users, meaning that a consumer group can contain multiple users, and a user can only belong to one consumer group.

4.10.5.3 delete

grammar

```
DROP CONSUMER GROUP [vc_name.]<group_ name>;
```

Example

```
DROP CONSUMER GROUP group1;
```

 **be careful**

If the consumer group is referenced in directive, the delete consumer group operation cannot be performed. It is necessary to delete the directive and disassociate the consumer group before deleting the consumer group.

4.10.6 Creating and managing Resource Pools

Resource pools are resource providers and managers in the execution process of cluster tasks, divided into static resource pools and dynamic resource pools. Static resource pools are resource providers, while dynamic resource pools are resource managers, which constrain the use of resources by tasks.

4.10.6.1 establish

grammar

```
CREATE RESOURCE POOL [vc_name.]<resource_ pool_ name> (pool_attribute=value
[, ...]) TYPE {static|dynamic} [BASE ON <parent_pool_name>]
```

Among them, base on<parent_ pool_ Used when creating dynamic resource pools, parent_ pool_ Name is the name of the static resource pool to which the dynamic resource pool belongs. pool_ The value of an attribute can be:

[priority={1|2|3|4|5|6|7|8 }]

< cpu_ percent=integer >

< max_ memory=integer >

< max_ temp_ diskspace=integer >

< max_ disk_ space=integer >

< max_ disk_ writeio= integer >

< max_ disk_ readio=integer >

[max_activetask=integer]

[task_max_parallel_degree=integer]

[task_waiting_timeout=integer]

[task_running_timeout=integer]

Table 483 Parameter Description

Parameter Name	explain
priority	1. Priority, divided into 8 levels, with 1 being the highest and 8 being the lowest, considered as a reserved parameter. It is recommended to configure it uniformly as 1; 2. This priority is only set for the CPU.
cpu_ percent	1. The percentage of CPU resources used, expressed

Parameter Name	explain
	as an integer, with a range of [1100]; 2. For static resource pools, CPU percentage bandwidth control is used (refer to the description of the CPU.cfs_quota_us parameter in Linux cgroup/CPU), and the calculation formula is: CPU.cfs_quota_us=(CPU_cores * CPU. cfs_seriod_us) * CPU_ percent ）； 3. For dynamic pools, weight control is used for CPUs (refer to the description of the CPU. shares parameter in Linux/cgroup, and the calculation formula is CPU. shares=1024 * CPU_percent).
max_ memory	1. The upper limit of the total usage of operator buffer, set in M. The total value of the dynamic pool should be less than or equal to the static pool setting where it is located; 2. If a single SQL statement has multiple operators and multiple SQL statements are executed concurrently, and the total operator buffer usage exceeds the total memory configured by the resource pool, then subsequent SQL tasks will report errors for processing; 3. If memory is not controlled, it is recommended to set a value greater than the sum of (physical memory+SWAP).
max_ temp_ diskspace	1. The amount of temporary disks that can be used during the execution of a single task in the pool, set in M; 2. If the temporary disk size is not controlled, it i s recommended to set a value greater than the physical disk space size; 3. This parameter is mandatory and the total number

Parameter Name	explain
	of dynamic pool parameters cannot exceed the static pool parameters.
max_ disk_ space	1. The total disk space occupied by all users associated with this resource pool, set in M; 2. If you do not control disk space, it is recommended to set a value greater than the physical disk space size; 3. This parameter is mandatory and the total number of dynamic pool parameters set cannot exceed the static pool parameters.
max_ disk_ writeio	1. The write rate limit for all disk access by tasks in the pool, set in MB/S, must be synchronized with the DC I/O control parameter _ gbase_ dc_ sync_ Only when used in conjunction with size can it take effect; 2. gbase_ dc_ sync_ Size must be less than or equal to the size set in all dynamic pools max_ disk_ Writeio value, recommended_ gbase_ dc_ sync_ Size=1M (For a detailed introduction to DC synchronous I/O control, please refer to the explanation section in this section); 3. If this item is not controlled, it is recommended to set the value to be greater than the theoretical value of physical disk write performance; 4. Set the total value of the dynamic pool to be less than or equal to its static pool setting.
max_ disk_ readio	1. The read rate limit for all disk access by tasks in the pool, set in MB/S; 2. If this item is not controlled, it is recommended to set a value greater than the theoretical val

Parameter Name	explain
	ue of physical disk read performance; 3. The total value of the dynamic pool should be less than or equal to its static pool setting.
max_ activetask	1. This parameter is exclusive to the dynamic resource pool, indicating the number of concurrent tasks in the pool. The default value is 20, which can be set based on memory. The memory usage limit for each task is max_ memory / max_ activetask; 2. If this parameter is too large, it will cause a decrease in memory usage for each task, resulting in task execution failure; 3. If the number of tasks issued is greater than this parameter value, the extra tasks will enter the waiting queue.
task_ max_ parallel_ degree	1. The concurrency of task execution in the pool defaults to 0. The resource pool does not control the size of concurrency, which is determined by the cluster parameter gbase_ parallel_ Degree control (refer to 5.3.1.2.1.2Gnode's concurrency control parameters). 2. The maximum value must be less than gbase_ parallel_ max_ thread_ in_ The value set for the pool parameter. 3. Note: One pass group, parallel hash group, parallel update, and parallel order by will occupy twice the parallelism during the parallel materialization phase. More than twice the parallelism should be allocated to avoid serialization. This parallelism does not affect the parallelism setting for loading.
task_ waiting_ timeout	1. The timeout time for waiting for execution of tasks in the pool, set in seconds, and its experience value

Parameter Name	explain
	is task_ waiting_ Timeout=maximum tolerance waiting queue length * (task_running_timeout * adjustment coefficient); 2. Due to task_ running_ The timeout is generally higher than the actual execution time of tasks in the pool, so appropriate adjustments can be made, with a default of 2592000s.
task_ running_ timeout	1. The timeout time for task execution in the pool is set in seconds, which can be adjusted based on the average execution statistics of tasks in the cluster. The default value is 2592000s.

explain

- DC (Data Cell) synchronous I/O control through parameters_ gbase_ dc_ sync_ Size (in bytes) control, which executes synchronous write I/O operations when the DC size of the column data file that has not been flushed in the buffer exceeds the specified value. The default value of this parameter is 1TB, and I/O operations will be completed by the system cycle write back mechanism. The minimum value of this parameter is 0, indicating that each controlled SQL calls a synchronous I/O operation.
- Resource pool memory control through parameters_ gbase_ resmgr_ memctrl_ Bypass to control the allocation strategy of largeBuffer, with a default value of 1. When the parameter value is 1, set the upper limit of each LargeBuffer according to the buffer settings or automatic evaluation values of each operator on the single machine; When the parameter value is 0, the lower value of the resource pool setting (maximum memory usage max_memory/maximum number of tasks max_activetask) and the buffer setting (or automatic evaluation value) for each operator on a single machine is taken as the upper limit for each LargeBuffer.
- A static resource pool can contain multiple dynamic resource pools.
- A dynamic resource pool can only and must belong to a static resource pool.
- The dynamic pool and the static pool it belongs to must be within the same VC.
- The following parameters must be assigned when creating a static resource pool:

cpu_ percent

max_ memory

max_ temp_ diskspace

max_ disk_ space

max_ disk_ writeio

max_ disk_ readio

Due to other parameters being invalid for static resource pools, it is recommended not to appear in the definition statement.

- The following parameters must be assigned when creating a dynamic resource pool:

  cpu_ percent

  max_ memory

  max_ temp_ diskspace

  max_ disk_ space

  max_ disk_ writeio

  max_ disk_ readio

  The sum of dynamic resource pool parameters cannot exceed the corresponding parameters of the attached static pool. Other parameters can be selected.

- When creating a static resource pool, hardware resources are not detected, such as max_ disk_ Space=50G, and a disk size of only 30G can still create a resource pool.

- The sum of the corresponding parameter values in the dynamic resource pool must not be higher than the parameter value setting of the static resource pool to which it belongs. Dynamic pool CPUs of the same priority belonging to the same static resource pool_ The sum of percentages cannot be greater than 100%, and there is no limit to different priorities.

Example

- Create a static resource pool

```
create resource pool static_ pool0(
cpu_ percent=70,
max_ memory=1000,
max_ temp_ diskspace= 2000,
```

```
max_ disk_ space= 2000,
max_ disk_ writeio=1000,
max_ disk_ readio=1000) type static;
```

```
create resource pool static_ pool1(
cpu_ percent=30,
max_ memory=100,
max_ temp_ diskspace=200,
max_ disk_ space=200,
max_ disk_ writeio=100,
max_ disk_ readio=100) type static;
```

- Create a dynamic resource pool

```
create resource pool dynamic_ pool0(
priority=1,
cpu_ percent=100,
max_ memory=1000,
max_ temp_ diskspace=2000,
max_ disk_ space=2000,
max_ disk_ writeio=1000,
max_ disk_ readio=1000,
max_ activetask=2,
task_ max_ parallel_ degree=100,
task_ waiting_ timeout=100000,
task_ running_ timeout=100000)
TYPE dynamic BASE ON static_ pool0;
```

```
create resource pool dynamic_ pool1(
priority=2,
cpu_ percent=100,
max_ memory=90,
max_ temp_ diskspace=200,
max_ disk_ space=200,
max_ disk_ writeio=90,
max_ disk_ readio=90,
max_ activetask=20,
```

```
task_ max_ parallel_ degree=10,
task_ waiting_ timeout=1000,
task_ running_ timeout=1000)
TYPE dynamic BASE ON static_ pool1;
```

4.10.6.2 change

grammar

- change one's name

```
ALTER RESOURCE POOL [vc_name.]<resource_ pool_ name> RENAME [TO] <new_ resource_ pool_ name>;
```

- Change parameters

```
ALTER RESOURCE POOL [vc_name.]<resource_ pool_ name> SET (pool_attribute=value [, ...]);
```

Among them, pool_ The value of an attribute can be:

[priority={1|2|3|4|5|6|7|8 }]

[cpu_percent=integer]

[max_memory=integer]

[max_temp_diskspace=integer]

[max_disk_space=integer]

[max_disk_writeio= integer]

[max_disk_readio=integer]

[max_activetask=integer]

[task_max_parallel_degree=integer]

[task_waiting_timeout=integer]

[task_running_timeout=integer]

Example

- change one's name

```
ALTER RESOURCE POOL resource_ pool_ 1 RENAME resource_ pool_ 2;
```

- Change parameters

```
ALTER RESOURCE POOL resource_ pool_ 2 SET (cpu_percent=20);
```

4.10.6.3 delete

grammar

```
DROP RESOURCE POOL [vc_name.]<resource_ pool_ name>;
```

be careful

- To delete a static resource pool, you must first delete its attached dynamic resource pool.
- If a dynamic resource pool is referenced in directive, it cannot be deleted. Before the dynamic resource pool can be deleted, it is necessary to remove the direct and disassociate the dynamic resource pool.

Example

```
DROP RESOURCE POOL resource_ pool_ 2;
```

4.10.6.4 Operating System Compatibility Instructions

Due to the fact that the resource management function relies on the system service cgroup, the effectiveness of resource management parameters may vary depending on the cgroup services of different operating systems. This is specifically explained.

Table -484 Operating System Compatibility Description

Configuration Item	Type of Resource Pool Belonging to	SUSE 11 SPX Redhat 6.X	SUSE 12+	Redhat 7+	remarks
cpu_ percent	Static and dynamic	Not effective	take effect	take effect	The option is missing from Cgroup,/cgroup/cpu/cpu cfs_ quota_ us The dynamic resource pool is not affected.

Configuration Item	Type of Resource Pool Belonging to	SUSE 11 SPX Redhat 6.X	SUSE 12+	Redhat 7+	remarks
Priority	move	take effect	take effect	take effect	——
max_ memory	Static and dynamic	take effect	take effect	take effect	——
max_ temp_ diskspace	Static and dynamic	take effect	take effect	take effect	——
max_ disk_ writeio	Static and dynamic	Not effective	take effect	take effect	The option is missing from Cgroup,/cgroup/blkio.
max_ disk_ readio	Static and dynamic	Not effective	take effect	take effect	The option is missing from Cgroup,/cgroup/blkio.
max_ activetask	move	take effect	take effect	take effect	——
task_ max_ parallel_ degree	move	Effective conditions	Effective conditions	Effective conditions	Limited by the configuration of the gnode thread pool, max is required to take effect_ activetask* task_ max_ parallel_ degree < gbase_ parallel_ max_ thread_ in_ pool Note: This parameter has a

Configuration Item	Type of Resource Pool Belonging to	SUSE 11 SPX Redhat 6.X	SUSE 12+	Redhat 7+	remarks
					significant impact on priority.
task_ waiting_ timeout	move	take effect	take effect	take effect	——
task_ running_ timeout	move	Effective conditions	Effective conditions	Effective conditions	Due to the impact of SQL being interruptible, non interruptible SQL such as DDL and index updates will not work.

be careful

- Only the supported mainstream operating systems have been explained here, and other operating systems can be inferred by checking whether there are related components in the cgroup.
- The issue of "cgroups CPU quota scheduling causing system downtime" exists in Redhat/Centos versions 7.3 and below. GBase 8a MPP Cluster resource management in multi static pool scenarios poses a certain risk of triggering this issue. RedHat versions 7.3/SUSE12 and above have fixed this issue. Therefore, it is recommended that users who use multi static pool scenarios upgrade to an updated operating system version.

4.10.7 Create and manage Resource Plans

Resource Plan is a plan for planning the use of resources in a cluster according to a certain pattern, in which resource consumption groups are linked to a reasonable

resource pool to ensure more effective utilization of cluster resources.

4.10.7.1 establish

grammar

```
CREATE RESOURCE PLAN [vc_name.]<plan_ name> [COMMENT ='comment'];
```

Table -485 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional parameter. If left blank, the current default VC will be used.
plan_ name	Resource plan name.
comment	notes.

 **explain**

- Resource Plan names are unique within the scope of the VC they belong to, and resource plans in different VCs are allowed to have duplicate names.

Example

```
CREATE RESOURCE PLAN plan1 COMMENT = 'test plan1';
```

4.10.7.2 change

grammar

- change one's name

```
ALTER RESOURCE PLAN [vc_name.]<plan_ name> RENAME [TO] <new_ plan_ name>;
```

Table -486 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional parameter. If left blank, the current default VC will be used.
plan_ name	Resource plan name.
new_ plan_ name	Name of the new plan.

- Change Description

```
ALTER RESOURCE PLAN [vc_name.]<plan_ name> COMMENT ='comment';
```

Table -487 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional parameter. If left blank, the current default VC will be used.
plan_ name	Resource plan name.
comment	Updated annotation information.

Example

```
ALTER RESOURCE PLAN plan1 RENAME plan3;
ALTER RESOURCE PLAN plan3 COMMENT = 'test plan3';
```

4.10.7.3 delete

grammar

```
DROP RESOURCE PLAN [vc_name.]<plan_ name>;
```

Table -488 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional parameter. If left blank, the current default VC will be used.
plan_ name	Resource plan name.

be careful

If the resource plan is referenced in directive, it cannot be deleted. It is necessary to delete the directive to disassociate the resource plan before it can be deleted.

Example

```
DROP RESOURCE PLAN plan3;
```

4.10.8 Create Resource Directive

The Resource Directive specifies the linkage relationship between resource consumption groups and dynamic resource pools in the resource plan.

4.10.8.1 establish

grammar

```
CREATE RESOURCE DIRECTIVE [vc_name.]<directive_ name> (
        PLAN_ NAME = 'plan_ name',
        GROUP_ NAME = 'group_ name',
        POOL_ NAME = 'resource_ pool_ name',
        [COMMENT = 'comment']
);
```

Table -489 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional parameter. If left blank, the current default VC will be used.
directive_ name	Resource directive name.
PLAN_ NAME	Resource plan name.
GROUP_ NAME	Consumer group name.
POOL_ NAME	Resource pool name.
COMMENT	notes.

be careful

- The plan, pool, and group involved in the resource creation directive must have already been created.
- If there is an active plan for the current VC, it is not allowed to perform the create directive operation.
- A resource consumption group within the same resource plan can only be linked to one dynamic resource pool.
- A dynamic resource pool within the same resource plan can be linked to multiple resource consumption groups.
- Any planned default consumer group must be associated with a dynamic resource pool, which means creating a corresponding directive.

Example

```
CREATE RESOURCE DIRECTIVE vc1.directive1 (
        PLAN_ NAME = 'plan1',
        GROUP_ NAME ='group1',
         POOL_ NAME = 'dynamic_ pool0',
        COMMENT = 'test'
);

CREATE RESOURCE DIRECTIVE vc1.directive2 (
        PLAN_ NAME = 'plan1',
        GROUP_ NAME = 'default_ consumer_ group',
         POOL_ NAME = 'dynamic_ pool0',
        COMMENT = 'test'
);
```

4.10.8.2 delete

grammar

```
DROP RESOURCE DIRECTIVE [vc_name.]<directive_ name>;
```

Table -490 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional parameter. If left blank, the current default VC will be used.
directive_ name	Resource directive name.

be careful

If there is an active resource plan for the current VC, it is not allowed to perform the delete directive operation.

Example

```
DROP RESOURCE DIRECTIVE directive1;
```

4.10.9 Activate/disable resource management

4.10.9.1 activation

grammar

```
ACTIVE RESOURCE PLAN <resource_ plan_ name> ON VC <vc_ name>;
```

be careful

- Under the same VC, only one plan can be activated during the same time period;
- If it is the default VC, VC_ Name can be used to view the name of the default VC using show VCs;
- In any resource plan, the default consumer group must be associated with a dynamic pool in order to activate the resource plan.

Example

```
ACTIVE RESOURCE PLAN plan1 ON VC vc1;
```

4.10.9.2 close

grammar

```
DEACTIVE RESOURCE PLAN ON VC <vc_ name>;
```

Example

```
DEACTIVE RESOURCE PLAN ON VC vc1;
```

4.10.10 Resource Management Configuration Item Management

grammar

```
ALTER RESOURCE CONFIG [vc_name.]<config_ item_ name> = <config_ value>;
```

Table 491 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional parameter. If left blank, the current default VC will be used.
config_ item_ name	Configuration item name, currently valid configuration item names include gbase_ resource_ monit_ Record and GBase_ resource_ monit_ record_ Interval, both are global parameter configuration items. Among them: gbase_ resource_ monit_ Record=0/1 (1 indicates recording resource monitoring and statistical information, 0 indicates not recording, default value is 1). gbase_ resource_ monit_ record_ Interval sets the sampling interval for recording, with a default value of 300, in seconds.

4.10.11 Resource Management Information Query

4.10.11.1 Define Information Query

4.10.11.1.1 Query Consumer Group Definition

grammar

```
SELECT * FROM gbase.consumer_ group;
```

Example

```
gbase> SELECT * FROM gbase.consumer_ group;
+-------------------+---------------------+-------------+---------+
| consumer_ group_ id | consumer_ group_ name | comment         | vc_ id     |
+-------------------+---------------------+-------------+---------+
|                1900733 | group1                      | test group1 | vc00001 |
|                1900734 | group2                      | test group2 | vc00001 |
+-------------------+---------------------+-------------+---------+
2 rows in set (Elapsed: 00:00:00.00)
```

4.10.11.1.2 Query the association definition between consumer group and user

grammar

```
SELECT * FROM gbase.consumer_ group_ user;
```

Example

```
gbase> SELECT * FROM gbase.consumer_ group_ user;
+-------------------+-----------+
| consumer_ group_ id | user_ name |
+-------------------+-----------+
|             1900733 | user1     |
|             1900734 | user2     |
+-------------------+-----------+
2 rows in set (Elapsed: 00:00:00.00)
```

4.10.11.1.3 Query resource pool definition

grammar

```
SELECT * FROM gbase.resource_ pool;
```

Among them, gbase.resource_ Field max in the pool table_ memory， max_ tmp_ table_ space， max_ disk_ The unit of space is bytes.

Example

```
gbase> SELECT * FROM gbase.resource_ pool;
```

perhaps

```
gbase> SELECT * FROM gbase.resource_ pool\G
*************************** 1. row ***************************
          resource_ pool_ id: 1900737
        resource_ pool_ name: dynamic_ pool0
        resource_ pool_ type: 0
parent_ resource_ pool_ id: 1900735
                    priority: 1
                max_ memory: 1048576000
      max_ tmp_ table_ space: 2097152000
           max_ disk_ space: 2097152000
             task_ parallel: 100
          max_ task_ number: 2
                cpu_ percent: 100
```

```
              disk_ write_ bps: 1048576000
               disk_ read_ bps: 1048576000
             waiting_ timeout: 100000
             running_ timeout: 100000
                       vc_ id: vc00001
*************************** 2. row ***************************
          resource_ pool_ id: 1900738
        resource_ pool_ name: dynamic_ pool1
        resource_ pool_ type: 0
parent_ resource_ pool_ id: 1900736
                    priority: 2
                max_ memory: 94371840
      max_ tmp_ table_ space: 209715200
             max_ disk_ space: 209715200
               task_ parallel: 10
            max_ task_ number: 20
                 cpu_ percent: 100
             disk_ write_ bps: 94371840
               disk_ read_ bps: 94371840
             waiting_ timeout: 1000
             running_ timeout: 1000
                       vc_ id: vc00001
*************************** 3. row ***************************
          resource_ pool_ id: 1900735
        resource_ pool_ name: static_ pool0
        resource_ pool_ type: 1
parent_ resource_ pool_ id: 0
                    priority: 1
                max_ memory: 1048576000
      max_ tmp_ table_ space: 2097152000
             max_ disk_ space: 2097152000
               task_ parallel: 0
            max_ task_ number: 9223372036854775807
                 cpu_ percent: 100
             disk_ write_ bps: 1048576000
```

```
                disk_ read_ bps: 1048576000
              waiting_ timeout: 2592000
              running_ timeout: 2592000
                          vc_ id: vc00001
*************************** 4. row ***************************
           resource_ pool_ id: 1900736
        resource_ pool_ name: static_ pool1
        resource_ pool_ type: 1
parent_ resource_ pool_ id: 0
                    priority: 1
                  max_ memory: 104857600
      max_ tmp_ table_ space: 209715200
              max_ disk_ space: 209715200
                task_ parallel: 0
            max_ task_ number: 9223372036854775807
                 cpu_ percent: 50
              disk_ write_ bps: 104857600
                disk_ read_ bps: 104857600
            waiting_ timeout: 2592000
            running_ timeout: 2592000
                          vc_ id: vc00001
4 rows in set (Elapsed: 00:00:00.00)
```

4.10.11.1.4 Query resource plan definition

grammar

```
SELECT * FROM gbase.resource_ plan;
```

Example

```
gbase> SELECT * FROM gbase.resource_ plan;
+------------------+--------------------+---------+---------+
| resource_ plan_ id | resource_ plan_ name | comment | vc_ id      |
+------------------+--------------------+---------+---------+
|                1900732 | plan3                      | NULL        | vc00001 |
+------------------+--------------------+---------+---------+
```

```
1 row in set (Elapsed: 00:00:00.00)
```

4.10.11.1.5 Query resource directive definitions

grammar

```
SELECT * FROM gbase.resource_ plan_ directive;
```

Example

```
gbase> SELECT * FROM gbase.resource_ plan_ directive;
+-----------------------------+-----------------+------------------+-----------------+----------+---------+
| resource_ plan_ directive_ name | resource_ plan_ id | consumer_ group_ id | resource_ pool_ id |
comments | vc_ id     |
+-----------------------------+-----------------+------------------+-----------------+----------+---------+
| directive1                  |         1900732 |          1900733 |
1900737 | NULL     | vc00001 |
| directive2                  |         1900732 |          1900734 |
1900738 | NULL     | vc00001 |
| directive3                  |         1900732 |                1 |
1900738 | NULL     | vc00001 |
+-----------------------------+-----------------+------------------+-----------------+----------+---------+
3 rows in set (Elapsed: 00:00:00.00)
```

4.10.11.2 Query of runtime information

4.10.11.2.1 Query active plan settings

grammar

```
SELECT * FROM gbase.resource_ config;
```

Example

```
gbase> SELECT * FROM gbase.resource_ config;
+--------------------------------------+-------------+------------------+------------------+---------+
| config_ name                         | config_ type | config_ int_ value | config_ str_ value |
vc_ id     |
+--------------------------------------+-------------+------------------+------------------+---------+
```

```
| active_ resource_ plan_ id                          | int                 |                          104 | NULL
| vc00002 |
| gbase_ resource_ monit_ record_ interval | int                      |                          300 | NULL
| vc00002 |
| gbase_ resource_ monit_ record                      | int                 |                            1 | NULL
| vc00002 |
+--------------------------------------+-------------+-----------------+-----------------+---------+
3 rows in set (Elapsed: 00:00:00.00)
```

Among them, the configuration item is active_ resource_ plan_ Configuration value of id config_ int_ Value is the ID of the currently activated resource plan, which can be found in the system table gbase.resource based on this ID_ Found the name of the currently active resource plan in the plan.

4.10.11.2.2 Viewing Tasks in Dynamic Resource Pools

grammar

```
SELECT * FROM information_ schema.processlist WHERE resource_ pool_ name =’ pool_ name’  AND vc=’vc_ name’;
```

The following auxiliary query conditions can be added, such as:

running_ time = 0; // Waiting for tasks

Example

```
gbase> SELECT * FROM information_ schema.processlist\G
*************************** 1. row ***************************
                    ID: 524
                TASKID: 0
            SUBTASKID: 0
              THREADID: 27957
                  USER: root
                  HOST: 192.168.6.154:27779
                    VC: vcname000001
                    DB: NULL
               COMMAND: Sleep
```

```
START_ TIME: 2018-07-06 14:39:43
TIME: 612
STATE:
RESOURCE_ POOL_ NAME: NULL
RESOURCE_ POOL_ ID: NULL
RESOURCE_ POOl_ PRIORITY: NULL
WAITING_ TIME: NULL
RUNNING_ TIME: NULL
LOCK: NULL
WAIT: NULL
INFO: NULL
TRACE: NULL
```

4.10.11.2.3 Viewing the Resource Pool's Resource Usage

grammar

```
SHOW RESOURCE POOL USAGE ON {coordinators | nodes} WHERE resource_ pool_ name = 'pool_ name' AND vc_ name=’ xxxx’ ;
```

Example

```
gbase> SHOW RESOURCE POOL USAGE ON coordinators WHERE resource_ pool_ name = 'pool1' AND vc_ name='vcname000001';
+--------------+------------------+--------------------+----------+---------------+---------------+---------+---------
------+
| NODE_ NAME         | RESOURCE_ POOL_ ID | RESOURCE_ POOL_ NAME | PRIORITY |
WAITING_ TASKS | RUNNING_ TASKS | VC_ ID     | VC_ NAME          |
+--------------+------------------+--------------------+----------+---------------+---------------+---------+---------
------+
| coordinator1 |                  1900741 | pool1                        |              1 |                     0 |
0 | vc00001 | vcname000001 |
| coordinator2 |                  1900741 | pool1                        |              1 |                     0 |
0 | vc00001 | vcname000001 |
| coordinator3 |                  1900741 | pool1                        |              1 |                     0 |
0 | vc00001 | vcname000001 |
+--------------+------------------+--------------------+----------+---------------+---------------+---------+---------
------+
3 rows in set (Elapsed: 00:00:00.01)
```

```
gbase> SHOW RESOURCE POOL USAGE ON nodes WHERE resource_ pool_ name = 'pool1' AND vc_ name='vcname000001';
+-----------+---------+--------------+------------------+--------------------+----------+---------------+------------
---+-----------+-----------+------------+--------------+-------------+
| NODE_ NAME | VC_ ID      | VC_ NAME             | RESOURCE_ POOL_ ID | RESOURCE_
POOL_ NAME | PRIORITY | RUNNING_ TASKS | WAITING_ TASKS | CPU_ USAGE | MEM_
USAGE | DISK_ USAGE | DISK_ WRITEIO | DISK_ READIO |
+-----------+---------+--------------+------------------+--------------------+----------+---------------+------------
---+-----------+-----------+------------+--------------+-------------+
| node1          | vc00001 | vcname000001 |                  1900741 | pool1                        |              1 |
```

```
0 |                0 |           0 |           0 |           0 |                0 |              0 |
| node2        | vc00001 | vcname000001 |             1900741 | pool1                |            1 |
0 |                0 |           0 |           0 |           0 |                0 |              0 |
| node3        | vc00001 | vcname000001 |             1900741 | pool1                |            1 |
0 |                0 |           0 |           0 |           0 |                0 |              0 |
+-----------+---------+--------------+-----------------+--------------------+----------+---------------+------------
---+-----------+-----------+------------+--------------+-------------+
3 rows in set (Elapsed: 00:00:01.82)
```

4.10.11.3 Statistical information query

4.10.11.3.1 View resource usage history information of dynamic resource pool tasks running

grammar

```
SHOW RESOURCE POOL STATUS ON {coordinators | nodes} WHERE resource_ pool_ name = 'pool_ name' AND vc_ name=’ xxxx’ ;
```

Example

```
gbase> SHOW RESOURCE POOL STATUS ON coordinators WHERE resource_ pool_ name = 'pool1' and vc_ name='vcname000001';
+--------------+-----------------+--------------------+----------+---------------+-----------------+----------------
--+---------------------+---------+--------------+
| NODE_ NAME         | RESOURCE_ POOL_ ID | RESOURCE_ POOL_ NAME | PRIORITY |
SERVIED_ TASKS | WAITING_ AVG_ TIME | RUNNING_ AVG_ TIME | SAMPLE_ TIME
| VC_ ID     | VC_ NAME          |
+--------------+-----------------+--------------------+----------+---------------+-----------------+----------------
--+---------------------+---------+--------------+
| coordinator1 |                65561 | pool1                |            2 |                0 |
0 |                0 | 2018-07-04 17:38:02 | vc00001 | vcname000001 |
| coordinator1 |                65576 | pool1                |            2 |                0 |
0 |                0 | 2018-07-04 17:38:24 | vc00001 | vcname000001 |
| coordinator1 |                65585 | pool1                |            2 |                0 |
0 |                0 | 2018-07-04 17:38:45 | vc00001 | vcname000001 |
| coordinator1 |                65594 | pool1                |            2 |                0 |
```

```
0 |                    0 | 2018-07-04 17:39:06 | vc00001 | vcname000001 |
| coordinator1 |                65711 | pool1                |            1 |                    6 |
0 |                    0 | 2018-07-04 17:39:26 | vc00001 | vcname000001 |
| coordinator1 |               458778 | pool1                |            2 |                    0 |
0 |                    0 | 2018-07-05 09:29:00 | vc00001 | vcname000001 |
| coordinator1 |               458791 | pool1                |            2 |                    0 |
0 |                    0 | 2018-07-05 09:29:22 | vc00001 | vcname000001 |
| coordinator1 |               458800 | pool1                |            2 |                    0 |
0 |                    0 | 2018-07-05 09:29:44 | vc00001 | vcname000001 |
| coordinator1 |               458809 | pool1                |            2 |                    0 |
0 |                    0 | 2018-07-05 09:30:05 | vc00001 | vcname000001 |
| coordinator1 |               458926 | pool1                |            1 |                    6 |
0 |                    0 | 2018-07-05 09:30:22 | vc00001 | vcname000001 |
```

gbase> SHOW RESOURCE POOL STATUS ON nodes WHERE resource_ pool_ name = 'pool1' AND vc_ name='vcname000001';

```
+-----------+---------+--------------+-----------------+--------------------+----------+---------------+------------
---+-------------+-----------+-----------+-------------+-------------+--------------------+
| NODE_ NAME | VC_ ID      | VC_ NAME            | RESOURCE_ POOL_ ID | RESOURCE_
POOL_ NAME | PRIORITY | RUNNING_ TASKS | WAITING_ TASKS | CPU_ USAGE      |
MEM_ USAGE | DISK_ USAGE | DISK_ WRITEIO | DISK_ READIO | SAMPLE_ TIME
|
+-----------+---------+--------------+-----------------+--------------------+----------+---------------+------------
---+-------------+-----------+-----------+-------------+-------------+--------------------+
| node1        | vc00001 | vcname000001 |               65561 | pool1                |            2 |
0 |                0 |              0 |            0 |              0 |                0 |                0 |
2018-07-04 17:38:07 |
| node1        | vc00001 | vcname000001 |               65576 | pool1                |            2 |
0 |                0 |              0 |            0 |              0 |                0 |                0 |
2018-07-04 17:38:29 |
| node1        | vc00001 | vcname000001 |               65585 | pool1                |            2 |
0 |                0 |              0 |            0 |              0 |                0 |                0 |
2018-07-04 17:38:51 |
| node1        | vc00001 | vcname000001 |               65594 | pool1                |            2 |
0 |                0 |              0 |            0 |              0 |                0 |                0 |
```

```
2018-07-04 17:39:12 |
| node1       | vc00001 | vcname000001 |              65711 | pool1                |           1 |
0 |                  0 |    104116128 |           0 |           0 |           0 |          24576 |
2018-07-04 17:39:31 |
| node1       | vc00001 | vcname000001 |             458778 | pool1                |           2 |
0 |                  0 |           0 |           0 |           0 |           0 |              0 |
2018-07-05 09:29:05 |
```

4.10.11.3.2 Viewing Dynamic Resource Pool Resource Control Events

grammar

```
SHOW RESOURCE POOL EVENTS WHERE resource_ pool_ name = 'pool_ name' AND vc_ name=’ xxxx’ ;
```

Example

```
gbase> SHOW RESOURCE POOL EVENTS WHERE resource_ pool_ name = 'pool1' AND vc_ name='vcname000001';
+--------------+---------+--------------+------------------+--------------------+----------------------+---------+-----------------------------+------------+--------------------+
| NODE_ NAME       | VC_ ID     | VC_ NAME          | RESOURCE_ POOL_ ID | RESOURCE_ POOL_ NAME | EVENT_ TIME                    | TASK_ ID | STATEMENT | EVENT_ TYPE | EVENT_ DESCRIPTION |
+--------------+---------+--------------+------------------+--------------------+----------------------+---------+-----------------------------+------------+--------------------+
| coordinator1 | vc00001 | vcname000001 |           655548 | pool1                | 2018-07-05 09:48:26 |  655594 | insert into t1 select * from t | Terminate  | Exceed max runtime |
| coordinator1 | vc00001 | vcname000001 |           655548 | pool1                | 2018-07-05 09:48:47 |  655596 | insert into t select * from t  | Terminate  | Exceed max runtime |
| coordinator1 | vc00001 | vcname000001 |           655548 | pool1                | 2018-07-05 09:58:03 |  720897 | insert into t1 select * from t | Terminate  | Exceed max runtime |
| coordinator1 | vc00001 | vcname000001 |           655548 | pool1                | 2018-07-05 09:58:17 |  720899 | insert into t1 select * from t | Terminate  | Exceed max runtime |
| coordinator1 | vc00001 | vcname000001 |           655548 | pool1                | 2018-07-05 10:20:11 | 1048577 | insert into t1 select * from t | Terminate  | Exceed max runtime |
| coordinator1 | vc00001 | vcname000001 |           655548 | pool1                | 2018-07-05
```

```
10:44:49 | 1179649 | insert into t1 select * from t | Terminate    | Exceed max runtime |
| coordinator1 | vc00001 | vcname000001 |                  655548 | pool1                          | 2018-07-05
10:45:48 | 1179655 | insert into t1 select * from t | Terminate    | Exceed max runtime |
+--------------+---------+--------------+------------------+--------------------+----------------------+---------+----
----------------------------+------------+--------------------+
7 rows in set (Elapsed: 00:00:00.02)
```

4.10.12 Resource Management Example

4.10.12.1 Loading and querying mixed scenarios

4.10.12.1.1 Background Introduction

Assuming there are two different users:

- UserLoad is a user who primarily loads tasks. During the day, there are generally low requirements for the number of tasks and urgency, while at night, a certain number of tasks are activated and they hope to complete the loading as soon as possible;
- UserSelect is a user with a focus on query services. During the day, there are many tasks that require timely response, while at night, the number of tasks is reduced.

This requires attaching different dynamic resource pools to the resource consumption groups of these two users during the day and at night, and continuously switching based on this pattern to achieve the goal of the same user being under different control at different time periods, thus using resources more reasonably.

4.10.12.1.2 Resource allocation plan

Due to the resource management limitations of GBase 8a MPP Cluster on memory only applied to the memory used by aggregation, connection, and other operators in the task, it is assumed that each Data node is reserved with 10G of memory allocated to the operator buffer. Based on the resource requirements, the resource pool allocated to these two users is:

Allocate 20% CPU and 2GB of memory in the dynamic resource pool corresponding to UserLoad during the day;

During the day, UserSelect allocates 80% CPU and 8GB of memory;

At night, UserLoad allocates 80% CPU and 8GB of memory;

At night, UserSelect allocates 20% CPU and 2GB of memory.

4.10.12.1.3 Implementation steps

Two plans can be created to allocate resources for different time periods of day and night, and this resource control method can be achieved by switching plans.

步骤 1 Create resource consumption groups and hook users to corresponding groups

```
create consumer group group_ load comment = 'users for load';
create consumer group group_ select comment = 'users for select';
alter consumer group group_ load add user userload;
alter consumer group group_ select add user userselect;
```

步骤 2 Create Resource Pool

1. Static resource pool

```
create resource pool static_ pool0(
cpu_ percent=100,
max_ memory=10000,
max_ temp_ diskspace=10000,
max_ disk_ space=10000,
max_ disk_ writeio=1000,
max_ disk_ readio=1000) type static;
```

2. Dynamic resource pool

```
create resource pool high_ pool(
priority=1,
cpu_ percent=80,
max_ memory=8000,
max_ temp_ diskspace=5000,
max_ disk_ space=5000,
max_ disk_ writeio=600,
max_ disk_ readio=600,
max_ activetask=200,
task_ max_ parallel_ degree=100,
task_ waiting_ timeout=100000,
task_ running_ timeout=100000)
type dynamic base on static_ pool0;
```

```
create resource pool low_ pool(
priority=1,
cpu_ percent=20,
max_ memory=2000,
max_ temp_ diskspace=5000,
max_ disk_ space=5000,
max_ disk_ writeio=400,
max_ disk_ readio=400,
max_ activetask=200,
task_ max_ parallel_ degree=100,
task_ waiting_ timeout=100000,
task_ running_ timeout=100000)
type dynamic base on static_ pool0;
```

步骤 3 Create Resource Plan

```
create resource plan    plan_ day    comment = 'day plan';
create resource plan    plan_ night    comment = 'night plan';
```

步骤 4 Create Resource Instruction Plan

```
create resource directive directive1
(plan_name ='plan_day',
  pool_ name='high_ pool',
  group_ name = 'group_ select',
  comment = 'select user resource usage on day');

create resource directive directive2
(plan_name ='plan_day',
  pool_ name = 'low_ pool',
  group_ name = 'group_ load',
  comment = 'load user resource usage on day');

create resource directive directive3
(plan_name = ‘plan_day',
pool_ name='high_ pool',
group_ name = 'default_ consumer_ group',
comment = 'other user resource usage on day ');
```

```
create resource directive directive4
(plan_name='plan_night',
pool_ name='high_ pool',
group_ name = 'group_ load',
comment = ' load user resource usage on night ');

create resource directive directive5
( plan_name='plan_night',
   pool_ name='low_ pool',
   group_ name = 'group_ select',
   comment = ' select user resource usage on night ');

create resource directive directive6
(plan_name ='plan_night',
 pool_ name='low_ pool ',
 group_ name = 'default_ consumer_ group',
 comment = 'other user resource usage on night');
```

步骤 5 Activate plan

- During the day (assuming 8:00 am)

```
active resource plan plan_ day on vc vc1;
```

- Evening (assuming 20:00 pm)

```
deactive resource plan on vc vc1;
active resource plan plan_ night on vc vc1;
```

4.10.12.2 Running queries during the day and batch scenarios at night

This scenario is consistent with the above example scenario, as long as the resource plan is divided according to users, it can be implemented in the above way.

4.10.12.3 High and low speed limit group scenarios

1. This scenario is consistent with the previous example scenario, but the difference lies in the setting of disk read/write I/O values, such as limiting the resource plan for low write speed limit groups (such as 10M/S) and high write speed limit groups (such as 100M/S).
2. Due to kernel defects in lower versions of operating systems such as RH6.x/SUSE11, high and low speed groups exhibit serialization on ext log file systems, with high speed groups only reaching up to twice the peak of low speed groups. It is recommended to use the I/O speed limit function in operating systems above RH7.3/SUSE12.

4.10.12.4 Full day data processing and query Task parallelism parallel scenario

4.10.12.4.1 Background Introduction

Data processing users: UserA, UserB;
Query users: UserC, UserD, UserE;
Other users are processed according to the data processing user method.
Resource allocation requirements: Ensure query performance and control processing consumption performance.

4.10.12.4.2 Implementation steps

步骤 1 Create resource consumption groups and associate users

```
create consumer group group_ process    comment = 'users for process';
```

```
create consumer group group_ select    comment = 'users for select';
alter consumer group group_ process    add user usera;
alter consumer group group_ process    add user userb;
alter consumer group group_ select    add user userc;
alter consumer group group_ select    add user userd;
alter consumer group group_ select    add user usere;
```

步骤 2 Create Resource Pool

1. Static resource pool

```
create resource pool static_ pool0(
 cpu_ percent=100,
 max_ memory=10000,
 max_ temp_ diskspace=10000,
 max_ disk_ space=10000,
max_ disk_ writeio=1000,
 max_ disk_ readio=1000) type static;
```

2. Dynamic resource pool

```
create resource pool pool_ select(
priority=1,
cpu_ percent=80,
max_ memory=6000,
max_ temp_ diskspace=5000,
max_ disk_ space=5000,
max_ disk_ writeio=600,
max_ disk_ readio=600,
max_ activetask=200,
task_ max_ parallel_ degree=100,
task_ waiting_ timeout=100000,
task_ running_ timeout=100000)
type dynamic base on static_ pool0;

create resource pool pool_ process(
priority=1,
cpu_ percent=20,
max_ memory=4000,
```

```
max_ temp_ diskspace=5000,
max_ disk_ space=5000,
max_ disk_ writeio=400,
max_ disk_ readio=400,
max_ activetask=200,
task_ max_ parallel_ degree=100,
task_ waiting_ timeout=100000,
task_ running_ timeout=100000)
type dynamic base on static_ pool0;
```

步骤 3 Create Resource Plan

```
create resource plan    resource_ plan    comment = 'resource plan';
```

步骤 4 Create Resource Instruction Plan

```
create resource directive directive1
(plan_name = 'resource_plan' ,
  pool_ name = 'pool_ select',
  group_ name = 'group_ select',
  comment = 'select user resource usage ');

create resource directive directive2
(plan_name = 'resource_plan',
  pool_ name = 'pool_ process',
  group_ name = 'group_ process',
  comment = 'process user resource usage ');

create resource directive directive3
(plan_name = 'resource_plan',
  pool_ name = ' pool_ process ',
  group_ name = 'default_ consumer_ group',
  comment = 'other user resource usage ');
```

步骤 5 Activate Plan

```
active resource plan resource_ plan on vc vc1;
```

4.10.12.5 Support for advanced user spot check scenarios

4.10.12.5.1 Background Introduction

There are:

Data processing users: UserA, UserB;

Query users: UserC, UserD, UserE;

Spot check users: UserCheck;

It is required to randomly check the user's UserCheck query to obtain the highest priority and reserve memory and disk I/O resources for UserCheck.

4.10.12.5.2 Resource allocation plan

(1) Establish an exclusive dynamic resource pool for spot checking users;

(2) Set dynamic resource pool high priority and CPU_ Percentage is a higher value;

(3) Allocate necessary memory to spot check users.

4.10.12.5.3 Implementation steps

步骤 1 Create resource consumption groups and associate users

```
create consumer group group_ process    comment = 'users for process';
create consumer group group_ select    comment = 'users for select';
create consumer group group_ check    comment = 'users for check';

alter consumer group group_ check    add user usercheck;
alter consumer group group_ process    add user usera;
alter consumer group group_ process    add user userb;
alter consumer group group_ select    add user userc;
alter consumer group group_ select    add user userd;
alter consumer group group_ select    add user usere;
```

步骤 2 Create Resource Pool

```
create resource pool static_ pool0(
cpu_ percent=100,
max_ memory=10000,
max_ temp_ diskspace= 10000,
```

```
max_ disk_ space= 10000,
max_ disk_ writeio=1000,
max_ disk_ readio=1000) TYPE static;

create resource pool pool_ check(
priority=1,
cpu_ percent=99,
max_ memory=4000,
max_ temp_ diskspace=50000,
max_ disk_ space=50000,
max_ disk_ writeio=600,
max_ disk_ readio=600,
max_ activetask=200,
task_ max_ parallel_ degree=100,
task_ waiting_ timeout=100000,
task_ running_ timeout=100000)
type dynamic base on static_ pool0;

create resource pool pool_ select(
priority=3,
cpu_ percent=70,
max_ memory=4000,
max_ temp_ diskspace=5000,
max_ disk_ space=5000,
max_ disk_ writeio=200,
max_ disk_ readio=200,
max_ activetask=200,
task_ max_ parallel_ degree=100,
task_ waiting_ timeout=100000,
task_ running_ timeout=100000)
type dynamic base on static_ pool0;

create resource pool pool_ process(
priority=3,
cpu_ percent=30,
```

```
max_ memory=2000,
max_ temp_ diskspace=5000,
max_ disk_ space=5000,
max_ disk_ writeio=200,
max_ disk_ readio=200,
max_ activetask=200,
task_ max_ parallel_ degree=100,
task_ waiting_ timeout=100000,
task_ running_ timeout=100000)
type dynamic base on static_ pool0;
```

步骤 3 Create Resource Plan

```
create resource plan    resource_ plan    comment = 'resource plan';
```

步骤 4 Create Resource Instruction Plan

```
create resource directive directive1
(plan_name = 'resource_plan',
  pool_ name = 'pool_ select',
  group_ name = 'group_ select',
  comment = 'select user resource usage ');

create resource directive directive2
(plan_name = 'resource_plan',
  pool_ name = 'pool_ process',
  group_ name = 'group_ process',
  comment = 'process user resource usage ');

create resource directive directive3
(plan_name = 'resource_plan',,
pool_ name = 'pool_ check',
group_ name = 'group_ check',
  comment ='check user resource usage ');

create resource directive directive4
(plan_name = 'resource_plan',
  pool_ name = ' pool_ process ',
```

```
group_ name = 'default_ consumer_ group',
comment = 'other user resource usage ');
```

步骤 5 Activate Plan

```
active resource plan resource_ plan on vc vc1;
```

4.10.12.6 Multi tenant isolation scenario

4.10.12.6.1 Background Introduction

There are:

User group High: UserA;

User group Low: UserB;

Require high and low user groups to isolate CPU resources, with the high group accounting for 90% of CPU resources and the low group accounting for 10% of CPU resources, without sharing them with each other.

4.10.12.6.2 Resource allocation plan

(1) Establish two static resource pools to isolate the CPUs of the High and Low user groups;

(2) Establish a dynamic resource pool under each static resource pool.

4.10.12.6.3 Implementation steps

步骤 1 Create resource consumption groups and associate users

```
create consumer group group_ high    comment = 'users for high';
create consumer group group_ low    comment = 'users for low';
alter consumer group group_ high    add user usera;
alter consumer group group_ low    add user userb;
```

步骤 2 Create Resource Pool

```
create resource pool static_ pool_ high(
cpu_ percent=90,
max_ memory=10000,
max_ temp_ diskspace= 10000,
```

```
max_ disk_ space= 10000,
max_ disk_ writeio=1000,
max_ disk_ readio=1000) TYPE static;

create resource pool static_ pool_ low(
cpu_ percent=10,
max_ memory=10000,
max_ temp_ diskspace= 10000,
max_ disk_ space= 10000,
max_ disk_ writeio=1000,
max_ disk_ readio=1000) TYPE static;

create resource pool pool_ high(
priority=1,
cpu_ percent=100,
max_ memory=4000,
max_ temp_ diskspace=50000,
max_ disk_ space=50000,
max_ disk_ writeio=600,
max_ disk_ readio=600,
max_ activetask=200,
task_ max_ parallel_ degree=100,
task_ waiting_ timeout=100000,
task_ running_ timeout=100000)
type dynamic base on static_ pool_ high;

create resource pool pool_ low(
priority=1,
cpu_ percent=100,
max_ memory=4000,
max_ temp_ diskspace=50000,
max_ disk_ space=50000,
max_ disk_ writeio=600,
max_ disk_ readio=600,
max_ activetask=200,
```

```
task_ max_ parallel_ degree=100,
task_ waiting_ timeout=100000,
task_ running_ timeout=100000)
type dynamic base on static_ pool_ low;
```

步骤 3 Create Resource Plan

```
create resource plan    resource_ plan    comment = 'resource plan';
```

步骤 4 Create Resource Instruction Plan

```
create resource directive directive1
(plan_name = 'resource_plan',
 pool_ name = 'pool_ high',
 group_ name = 'group_ high',
 comment = 'high user resource usage ');

create resource directive directive2
(plan_name = 'resource_plan',
 pool_ name = 'pool_ low',
 group_ name = 'group_ low',
 comment = 'low user resource usage ');
```

步骤 5 Activate Plan

```
active resource plan resource_ plan on vc vc1;
```

be careful

- Due to kernel defects in lower versions of operating systems such as RH6.x/SUSE11, it may lead to operating system downtime under heavy loads.
- It is recommended to use multiple static resource pool control in operating systems above RH7.3/SUSE12.

4.10.13 matters needing attention

4.10.13.1 System cgconfig service

- Do not modify the configuration file of the cgconfig service;
- Do not arbitrarily enter the mounting point of the cgconfig subsystem and keep it in a state to avoid abnormal resource management functions;
- Do not enter the mount point of the cgconfig subsystem to perform operations;
- Do not manually restart or shut down the cgconfig service when there are no mounting exceptions in the cgconfig subsystem.

4.10.13.2 Controlled SQL

The design goal of GBase 8a MPP Cluster resource management is to effectively control resource consumption, avoid resource competition among tasks by setting resource limits and priorities for tasks, and ensure the use of high priority task resources. At the same time, the resource management function limits the management and control of specific types of SQL operations, avoiding the insufficient utilization of system resources caused by small and frequent SQL occupying tasks.

The types of controlled SQL are as follows:

Table -492 Description of Controlled SQL Types

classification	explain
query	Select query operation.
DML	Load operation/update index (hash, fulltext) operation/insert select operation/update operation/merge operation/delete data operation.
DDL	Create index (hash, fulltext) operation/create as select operation/create like operation.
CALL	The query, DML, and DDL controlled statements contained within the stored procedure during the call.

explain

- Due to the fact that DDL operations themselves do not respond to interrupts, the DDL operations (CREATE INDEX, ALT TABLE ADD INDEX, etc.) in the above controlled SQL operations do not support runtime timeout processing.

- Due to the inconsistency between the SQL tasks executed by the cluster service program and the SQL tasks distributed to the standalone service program during disassembly, there are differences. This implementation method may result in the following phenomena:

  The cluster level SQL task is not a controlled SQL, but there is a controlled SQL in the SQL task that has been disassembled and distributed to a single machine. This also leads to the phenomenon of uncontrollable SQL tasks in the cluster layer being controlled on the single machine layer during the resource monitoring query process. For example, perform a rebalance operation.

  This phenomenon is currently a normal phenomenon and is a cluster implementation mechanism.

- Except for DDLs in controlled SQL, other DDLs are not limited by task count management, such as CREATE TABLE, ALT TABLE, DROP TABLE, CREATE USER, etc.

4.10.13.3 High and low priority user usage constraints

Under the following usage scenarios, the process of reading DC by high priority users will be affected by low priority users, resulting in similar execution performance between high and low priority users:

1. Using the same degree of parallelism;
2. Simultaneously using the same (similar) SQL to access data from the same column of the same table, and I/O is the main time consuming of the SQL (if the operator calculation is the main time consuming of SQL, the impact will be reduced);
3. The data accessed is cold data (if some of the data is hot, the impact of reading DC data decreases as the proportion of hot data increases).

Adjusting parallelism can reduce the mutual influence between high and low priority users when conditions 2 and 3 are fully met.

4.11Data migration tool

4.11.1 File Format Description

4.11.1.1 Text Format

Text format without escape (format=3)

- Rows are separated by ' n' by default (the line separator in a single loaded text file must be one

  To), users can customize line separators;
- The content of any field cannot contain row separators;
- Use the specified field separator to separate fields;
- Use the specified string to represent a null value;
- Fields can be surrounded by delimiters, and importing supports all fields with delimiters and some fields with packages

  There are three types of enclosing symbols and no enclosing symbols;
- The field content cannot be escaped.

Fixed length text format (format=4)

- Rows are separated by ' n' by default (the line separator in a single loaded text file must be one

  To), users can customize line separators;
- The content of any field cannot contain row separators;
- The same fields in all rows are stored in a rectangular format. If the field data size is not enough for the field width

  Fill in with a space character;
- Use the specified string to represent a null value;
- Field content cannot be surrounded by delimiters or escaped.

4.11.1.2 Rules for Escape Characters

Use C-style escape to support visible and invisible characters. The escape rules for C are as follows:

Table -493 Explanation of C Escape Rules

Escape character	Hexadecimal encoding
'\a'	0x07
'\b'	0x08
'\e'	0x1b
'\f'	0x0c
'\n'	0x0a
'\r'	0x0d
'\t'	0x09
'\v'	0x0b
'\\'	0x5c
'\0'	0x00
'\xFF'	//The backslash is followed by x, followed by two valid hexadecimal numbers (0x00-0xff, ignoring case), and is escaped as a single byte of the corresponding hexadecimal number.

4.11.2 Using the Orato8a tool

4.11.2.1 Installation of Oracle client

4.11.2.1.1 summary

This chapter mainly explains how to install the Oracle client in the Linux operating system. The purpose of installing it is to use the Orato8a data extraction tool. Usually, we recommend using the Orato8a tool on a physical machine with an Oracle client installed.

explain

The Oracle client program is not provided in the installation package of the cluster, and the description in this chapter is for reference only.

4.11.2.1.2 Obtain installation files

The client installation package file for Oracle in the Linux operating system, usually the rpm package. The complete Oracle client installation package includes the following installation package files:

- oracle-instantclient11.2-basic-11.2.0.4.0-1.x86_ 64.rpm
- oracle-instantclient11.2-devel-11.2.0.4.0-1.x86_ 64.rpm
- oracle-instantclient11.2-sqlplus-11.2.0.1.0-1.x86_ 64.rpm

explain

The installation package file above can be downloaded from the official website of Oracle.

4.11.2.1.3 Create User

On a machine with a Linux operating system, first switch to root and create a new operating system user.

Example: Create an oracli user.

```
$ su
password:
# /usr/sbin/useradd oracli
# passwd oracli
Change the password of user oracli.
New password:
Reenter a new password:
Passwd: All authentication tokens have been successfully updated.
```

4.11.2.1.4 Create directory

After successfully creating the user, the next step is to create the required directory.

Example: Using root user to create a directory and modify the permissions of the directory.

```
su – root
password:
# mkdir -p /home/oracli/network/admin
```

```
# chown -R oracli:oralci /home/oracli
# chmod -R 755 /home/oracli
```

4.11.2.1.5 Copy and modify the tnsnames.ora file

Copy the tnsnames.ora file on the Oracle server to the directory/home/oralci/network/admin on the Oracle client machine.

Step 1

View tnsnames.ora on the oracle server machine (192.168.103.79)

```
ll /opt/oracle/product/OraHome/network/admin/tnsnames.ora
-rw-r-----. 1 oracle11g oracle11g 316 May 21  2015
/opt/oracle/product/OraHome/network/admin/tnsnames.ora
```

Step 2

Use the scp command to copy files on the client machine (192.168.103.88).

```
su – root
password:
# scp root/ 111111@192.168.103.79 :/opt/oracle/product/OraHome/network/admin/tnsnames.ora
/home/oracli/network/admin
root/ 111111@192.168.103.79 's password:
tnsnames.ora
```

Step 3

Due to the use of root user for scp copying, it is still necessary to modify the tnsnames.ora permission after the copy is completed.

```
# chown -R oracli:oracli /home/oracli
# chmod -R 755 /home/oracli
```

Step 4

Switch to oracli user and view tnsnames.ora content:

```
su – oracli
$ cat /home/oracli/network/admin/tnsnames.ora
# tnsnames.ora Network Configuration File: /opt/oracle/product/10g/network/admin/tnsnames.ora
# Generated by Oracle configuration tools.
```

```
ORCL =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = localhost)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = orcl)
    )
  )

EXTPROC_ CONNECTION_ DATA =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC1))
    )
    (CONNECT_DATA =
      (SID = PLSExtProc)
      (PRESENTATION = RO)
    )
  )
```

explain

- In tnsnames.ora, it is necessary to modify the localhost in HOST=localhost to the IP of the Oracle server, for example: 192.168.103.79. Port=1521 is the default port used by Oracle, and if there are any changes, they also need to be modified. SERVICE_ In NAME=orcl, orcl is the service name that needs to be used when logging in with sqlplus64.

- Use the vi command to modify the IP address, and after the modification is completed: wq save to exit. The PORT port uses the default 1521 port and does not need to be modified.

vi tnsnames.ora

```
# tnsnames.ora Network Configuration File:
/opt/oracle/product/10g/network/admin/tnsnames.ora
# Generated by Oracle configuration tools.

ORCL =
  (DESCRIPTION =
```

```
    (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.103.79)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = orcl)
    )
  )
```

be careful

Because tnsnames.ora is copied from the Oracle server, once the configuration of the file on the Oracle server changes, such as adding a service name or deleting a service name, it is recommended to copy it back to the client's/home/oracli/network/admin/path and modify the IP address value of HOST in each service name. If the port changes, also modify the port value of PORT.

4.11.2.1.6 Install the rpm package for the client

Using the root user to install rpm packages, due to the dependency relationship between rpm packages, the installation sequence of Oracle client rpm packages is as follows:

- oracle-instantclient11.2-basic-11.2.0.4.0-1.x86_ 64.rpm
- oracle-instantclient11.2-devel-11.2.0.4.0-1.x86_ 64.rpm
- oracle-instantclient11.2-sqlplus-11.2.0.1.0-1.x86_ 64.rpm

Example

Install rpm package

```
# rpm -ivh oracle-instantclient11.2-basic-11.2.0.4.0-1.x86_ 64.rpm
Preparing...                ########################################### [100%]
   1:oracle-instantclient-ba########################################### [100%]

# rpm -ivh oracle-instantclient11.2-sqlplus-11.2.0.1.0-1.x86_ 64.rpm
Preparing...                ########################################### [100%]
   1:oracle-instantclient-sq########################################### [100%]
```

```
# rpm -ivh oracle-instantclient11.2-devel-11.2.0.4.0-1.x86_ 64.rpm
Preparing...                ########################################### [100%]
   1:oracle-instantclient-de########################################### [100%]
```

4.11.2.1.7 Configure client users'. bash_ Profile file

Users also need to modify the configuration information of the ". bash_profile" file under the client user, and add the following configuration information to. bash_ In the profile file.

```
export ORACLE_ HOME=/home/oracli
export SQLPATH=/home/oracli/network/admin
export TNS_ ADMIN=/home/oracli/network/admin
export LD_ LIBRARY_ PATH=/usr/lib/oracle/11.2/client64/lib:$LD_ LIBRARY_ PATH
export PATH=$PATH:$ORACLE_ HOME:$LD_ LIBRARY_ PATH
```

Example

```
$ cd /home/oracli
$ vi .bash_ profile

# .bash_ profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin

export PATH
```

```
export ORACLE_ HOME=/home/oracli
export SQLPATH=/home/oracli/network/admin
export TNS_ ADMIN=/home/oracli/network/admin
export LD_ LIBRARY_ PATH=/usr/lib/oracle/11.2/client64/lib:$LD_ LIBRARY_ PATH
export PATH=$PATH:$ORACLE_ HOME:$LD_ LIBRARY_ PATH
~
~
~
. bash_profile "20L, 488C written
```

After saving and exiting, use the source command to make the configuration file effective.

$ source .bash_ profile

4.11.2.1.8 Test connection to Oracle

After all installation and configuration work is completed, you can test whether the Oracle client can connect.

- Oracle server IP: 192.168.103.79;
- The login user and password for Oracle are both ct1;
- Oracle's service name: orcl;
- Connect to Oracle using the sqlplus64 command.

Example

```
sqlplus64 /nolog

SQL*Plus: Release 10.2.0.5.0 - Production on Fri Oct 18 11:18:04 2013

Copyright (c) 1982, 2010, Oracle.  All Rights Reserved.

SQL> conn ct1/ct1@//192.168.103.79/orcl
Connected.
```

4.11.2.2 Orato8a Tool Installation

Orato8a is a standalone data extraction tool that needs to be deployed on a machine that can access Oracle, or directly on the same server as the Oracle server.

4.11.2.2.1 Installation File Description

The Orato8a installation package is provided in a compressed form of tar. bz2. F or example: orato8a_ 26794_ Redhat6.2_ x86_ 64.tar.bz2。

explain

The specific meanings of each part of the installation package file name are as follows:

- Orato8a: represents the name of the tool;
- 26794: represents the version number of the tool;
- Redhat 6.2: represents that the operating system running this tool is Red hat 6.2;
- x86_ 64: indicates that the tool is a tool running on a 64 bit operating system.

4.11.2.2.2 Obtain installation files and unzip for installation

We take the 64 bit Linux operating system as an example to introduce to users h ow to extract and install Orato8a after obtaining it.

Step 1

Insert the installation CD into the CD drive and use the mount command to mou nt the CD drive to the Linux file system. The command to load the optical drive is as follows:

```
# mkdir /mnt/cdrom
# mount /dev/cdrom /mnt/cdrom
```

explain

Usually, loading an optical drive requires root user.

Step 2

Copy the compressed file of the installation package in the optical drive (such as orato8a_26794_Redhat6.2_x86_64. tar. bz2) to a directory on the file system. Ent er the directory (assuming the installation package is placed in the/root directory) and use the tar command in command line mode to decompress. The decompressi on command is as follows:

```
# tar xfj orato8a_ 26794_ Redhat6.2_ x86_ 64.tar.bz2
```

Step 3

After decompression, an orato8a executable program file will be generated in the decompression directory.

```
$ ll
Total usage 2068
……
-Rw-r - r -1 root 1380535 August 23 01:08 orato8a
-Rw-r - r -1 root 663929 August 22nd 17:13 orato8a_ 26794_ Redhat6.2_ x86_ 64.tar.bz2
```

4.11.2.2.3 Grammar format

grammar

```
./orato8a parameter_ 1 parameter_ 2 …… parameter_ n
```

explain

- parameter_ 1: The parameters of Orato8a can be used multiple times after an Orato8a. The parameters can be in full name format or abbreviated format;
- Full name format: -- parameter_ 1=parameter value, "=" cannot have spaces on either side;
- Abbreviation format: - parameter_ When using abbreviated format for parameter values, there should be no spaces between the parameters and their values;
- The user executing Orato8a must be a user who can access the oracle database.

Example

How to extract data from Oracle using the Orato8a tool? In this example, first log in to Oracle and then write a query SQL that queries 10 data items in the table lineorder in the database; Exit Oracle and use the Orato8a tool to extract query SQL result data to verify the data extraction function of Orato8a, as follows:

```
$ sqlplus /nolog

SQL*Plus: Release 11.2.0.1.0 Production on Thu Sep 26 16:37:41 2013

Copyright (c) 1982, 2009, Oracle.  All rights reserved.

SQL> conn ct1/ct1@//192.168.103.79/orcl
Connected.

SQL> CREATE TABLE lineorder_ test (
```

```
  2  lo_ orderkey            number(18),
  3  lo_ linenumber          number(18)
  4  );

Table created.

SQL> INSERT INTO lineorder_ test (lo_orderkey,lo_linenumber) VALUES(1,1);

1 row created.

SQL> INSERT INTO lineorder_ test (lo_orderkey,lo_linenumber) VALUES(1,2);

1 row created.

SQL> INSERT INTO lineorder_ test (lo_orderkey,lo_linenumber) VALUES(1,3);

1 row created.

SQL> INSERT INTO lineorder_ test (lo_orderkey,lo_linenumber) VALUES(2,1);

1 row created.

SQL> INSERT INTO lineorder_ test (lo_orderkey,lo_linenumber) VALUES(2,2);

1 row created.

SQL> INSERT INTO lineorder_ test (lo_orderkey,lo_linenumber) VALUES(2,3);

1 row created.

SQL> INSERT INTO lineorder_ test (lo_orderkey,lo_linenumber) VALUES(2,4);

1 row created.

SQL> INSERT INTO lineorder_ test (lo_orderkey,lo_linenumber) VALUES(3,1);

1 row created.

SQL> INSERT INTO lineorder_ test (lo_orderkey,lo_linenumber) VALUES(3,2);

1 row created.

SQL> INSERT INTO lineorder_ test (lo_orderkey,lo_linenumber) VALUES(3,3);
```

```
1 row created.

SQL> COMMIT;

COMMIT complete.

SQL> SELECT LO_ ORDERKEY, LO_ LINENUMBER FROM lineorder_ test;

LO_ ORDERKEY LO_ LINENUMBER
----------- -------------
          1             1
          1             2
          1             3
          2             1
          2             2
          2             3
          2             4
          3             1
          3             2
          3             3

10 rows selected.
```

Then exit Oracle and use Orato8a to extract the data queried above.

```
$ ./ orato8a --user='ct1/ ct1ct1@orcl ' --query="select LO_ORDERKEY, LO_LINENUMBER
FROM lineorder_test" --file='/opt/orato8a_ output/lineorder.txt ' --field=";" --format=3
export columns: 2
export rows: 10
export time: 0 sec
process ok!
$ cat /opt/orato8a_ output/lineorder.txt
1; one
1; two
1; three
2; one
2; two
2; three
2; four
3; one
3; two
3; three
```

The content in the exported data file is consistent with the query results in the Oracle system.

4.11.2.2.4 Parameter Description

4.11.2.2.4.1 user

function

Specify the relevant parameters for connecting to the oracle database, including user name, password, IP address of the server where the oracle database is located, oracle listening port number, and oracle instance name.

Format Grammar

User name [/password] [@ [Server IP of oracle database: oracle listening port number/] oracle instance name]

Example

Example 1

--user='orcl/ orcl@192.168.103.109 :1521/maya '

Example 2

-u'orcl/ orcl@192.168.103.109 :1521/maya'

Parameter Description

Table 494 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
user	u	nothing	nothing

explain

The username and password contain special characters, which can refer to the sqlplus rules:

- English single quotation marks (') are not supported;
- When containing special characters such as/, @, etc., the username or password needs to be enclosed in English double quotes before being used as a whole

Enclosed in English single quotation marks, examples are as follows:

create user "test/" identified by "pass@"

./orato8a --user='"test/"/"pass@"'@TEST [--owner_name="test/"] ...

- A username created using double quotation marks requires the use of a 'user' method, from the outside to the inside

Single quotation and double quotation marks, examples are as follows:

```
create user "test" identified by pass;

./orato8a --user='"test"/pass'@TEST [--owner_name='"test"']...
```

4.11.2.2.4.2 query

function

- Specify the query SQL statement used for exporting data. Due to the frequent occurrence of spaces in query statements, this parameter refers to

Timing requires double quotation marks to qualify. This parameter cannot be used together with the parallel parameter.

- The query statement should be an SQL that conforms to Oracle syntax and only returns a set of results.

Example

Example 1

```
--query="select * from lineorder"
```

Example 2

```
-q"select * from lineorder"
```

Parameter Description

Table -495 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
query	q	nothing	nothing

4.11.2.2.4.3 table_ name

function

Specify a table name, which cannot be used together with the query parameter.

Example

Example 1

```
--table_ name='lineorder'
```

Example 2

```
-t'lineorder'
```

Parameter Description

Table -496 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
table_ name	t	nothing	nothing

4.11.2.2.4.4 owner_ name

function

- When exporting a full table, specify the username under which the exported table belongs. This parameter cannot be the same as the query parameter

  When used;

- If this parameter is not specified, it defaults to the logged in user.

Example

Example 1

```
--owner_ name ='use1'
```

Example 2

```
--owner_ name ='use1'
```

Example 3

```
-o'user1'
```

Parameter Description

Table -497 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
owner_ name	o	nothing	log on user

4.11.2.2.4.5 file

function

Specify the storage path and file name for the exported data file.

explain

This parameter can be specified as a file name that contains an absolute path or a file name that contains a relative path. When this parameter is specified as a file name that does not contain a path, the file is saved in the current path.

Example

Example 1

```
--file='aaa.txt'

--file='/home/lina/aaa.txt'

--file='../ aaa.txt'
```

Example 2

```
-f'aaa.txt'

-f'/home/lina/aaa.txt'  .....

-f'../ aaa.txt'
```

Parameter Description

Table 498 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
file	f	nothing	nothing

4.11.2.2.4.6 format

function

Specify the data format for exporting data files.

explain

When set to 3, the exported data is in text format without escape.

Example

Example 1

```
--format='3'
```

Example 2

```
-m'3'
```

Parameter Description

Table -499 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
format	m	three	three

4.11.2.2.4.7 field

function

- Specify the field separator;
- Parameter values can be represented using characters themselves, escape characters, or hexadecimal notation;
- When exporting using format=3, the value of this parameter must be set.

Example

Example 1

```
--field=","

--field ="\x2c"

--field=" x'2c'"
```

Example 2

```
-e","

-e"\x2c"

-e"x'2c'"
```

explain:

Table -4100 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
field	e	Up to 15 characters	nothing

4.11.2.2.4.8 string_ qualifier

function

Specify the field delimiter, which is only valid when format=3. If a field bounding box is set, all fields will be added with a field bounding box. If there are duplicate parts between the field content and the field bounding box content, use the field bounding box to escape the field content.

Example

- Parameter values can be represented using characters themselves, escape characters, or hexadecimal notation.

Example 1

```
--string_ qualifier ='@'
```

Example 2

```
-s'@'
```

explain

-No spaces can be used between s and subsequent parameters.

- If the field delimiter is set to hexadecimal, the command parameter is followed by double quotes surrounding hexadecimal

Value.

Example 1

```
--string_ qualifier ="x'62'"
```

Example 2

```
-s"x'62'"
```

Parameter Description

Table -4101 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
string_ qualifier	s	One character	nothing

4.11.2.2.4.9 line_ separator

function

This parameter is used to set the row separator. This parameter is only valid when format=3.

Example

- Single byte line separator:

Example 1

```
--line_ separator ='@'
```

Example 2

```
-l'@'
```

- Multi byte line separator:

Example 3

```
--line_ separator='*|*'
```

Example 4

```
-l'*|*'
```

- If set to a hexadecimal line separator, the command parameter is followed by double quotes surrounding the hexadecimal value.

Example 5

```
--line_ separator ="x'6223'"
```

Example 6

```
-l"x'6223'"
```

Parameter Description

Table -4102 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
line_ separator	l	Up to 15 characters	'\n'

4.11.2.2.4.10 null_ value

function

This parameter is used to set a NULL value. This parameter is only valid when format=3.

Example

Example 1

```
--null_ value ='\\N'
```

Example 2

```
-n'\\N'
```

Parameter Description

Table 4103 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
null_ value	n	Maximum 15 characters	\N

4.11.2.2.4.11 parallel

function

Specify parallelism. In this mode, table needs to be used_ The name parameter specifies the table name, and the query parameter cannot be used, which means that using parallel mode can only export the entire table.

Example

Example 1

```
--parallel ='4'
```

Example 2

```
-T'2'
```

Parameter Description

Table -4104 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
parallel	T	1- Number of CPU cores	Default 4

4.11.2.2.4.12 blob_ conf

function

Specify the storage configuration for clob and blob fields. Clob and blob field data greater than 32kB will be stored on HBase or HDFS. The parameter value is the XML file path and is only valid when format=3. The content of the XML file is as follows:

```
<? xml version="1.0" encoding="UTF-8" ?>

<orato8a>

    <BlobStorage>

                        <HBase>

                            <Host>

                                <ServerName>192.168.10.114</ServerName>

                            <Port>9090</Port>

                                <UserName></UserName>

                            <Password></Password>

                            </Host>

                            <MaxConn>5</MaxConn>

                            <ConnRetryTime>3</ConnRetryTime>

                            <TableName>HbaseStream</TableName>

                            <ColFamilyName>file</ColFamilyName>

                            <MaxSize>16777216</MaxSize>

                        </HBase>
```

```
                    <HDFS>
                        <Host>
                            <ServerName>192.168.10.114</ServerName>
                            <Port>50070</Port>
                            <UserName>gbase</UserName>
                            <Password></Password>
                        </Host>
                        <Path>/blob</Path>
                    <MaxSize>17179869184</MaxSize>
                    </HDFS>
                    <Cache>
                        <Path>/tmp</Path>
                        <MaxSize>2147483648</MaxSize>
                    </Cache>
    </BlobStorage>
</orato8a>
```

explain

Table -4105 Configuration Item Description

Configuration Item	explain
HBase.Host	HBase host address, which can have multiple
HBase.MaxConn	HBase maximum number of links, default value: 5
HBase.ConnRetryTime	HBase link retry count, default value: 3
HBase.TableName	HBase table name **(requires pre creation)**
HBase.ColFamilyName	HBase Family Name
HBase.MaxSize	The maximum length of a blob stored in HBase. Files exceeding this length will be stored in HDFS, with a default value of 16777216 (16MB)
HDFS.Host	HDFS host address, which can have multiple

HDFS.Path	HDFS storage file path **(requires pre creation)**
HDFS.MaxSize	The maximum length of a blob stored in HDFS. If the file length exceeds this value, an export error will be reported. The default value is 17179869184 (16GB)
Cache.Path	BlobCache path **(requires pre creation)**
Cache.MaxSize	BlobCache maximum length, default value: 2147483648 (2GB)

Example

Example 1

```
--blob_ conf ='/home/gbase/orato8a/orato8a.xml'
```

Example 2

```
-c'/home/gbase/orato8a/orato8a.xml'
```

Parameter Description

Table 4106 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
blob_ conf	c	File Path	nothing

explain

Description of the storage location for clob or blob field data:

- When the data length is less than 32kB, the data is directly written to the exported data file;
- When the data length is greater than or equal to 32kB and less than or equal to HBase.MaxSize, the data is written to HBase, and the exported data file records the URL information of HBase records;
- When the data length is greater than HBase.MaxSize and less than or equal to HDFS.MaxSize, the data is written to the HDFS file, and the exported data file records the URL information of the HDFS file;
- When the data length is greater than HDFS.MaxSize, an export error is reported.

4.11.2.2.4.13 task_ number

function

Specify the task number. The task number should be unique within the cluster to prevent blob file name conflicts stored on HDFS.

Example

Example 1

```
--task_ number ='2'
```

Example 2

```
-g'2'
```

Parameter Description

Table -4107 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
task_ number	g	1-64 bit unsigned integer maximum	Default 0, automatically generated randomly

4.11.2.2.4.14 cipher

function

When the user specifies this parameter, orato8a converts the username and password specified by the user into an encrypted string and outputs it.

Example

```
./orato8a –uroot/ 1234@test -C or -- cipher
```

Parameter Description

Table -4108 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
cipher	C	nothing	nothing

4.11.2.2.4.15 use_ cipher

function

The user specifies this parameter to inform Orato8a of the username and password to use ciphertext.

Example

```
./orato8a - uC53C4EB43A65D4E6CF241A2F290578DF - L or -- use_ cipher
```

Parameter Description

Table -4109 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
use_ cipher	L	nothing	nothing

4.11.2.2.4.16 skip_ blob

function

Used to specify the processing method for blob or clob type columns.

explain

- 0: Indicates the normal extraction of blob and clob type column data and export to a file;
- 1: Indicates skipping the extraction of blob and clob type column data;
- 2: Represents extracting blob and clob type column data, but not writing it to a file.

Example

Example 1

```
./orato8a --skipblob=1
```

Example 2

```
./orato8a c1
```

Parameter Description

Table -4110 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
skip_ blob	c	0-2	0

4.11.2.2.4.17 encoding

function

Used to specify the encoding method for blob or clob type column data.

explain

- Text: Refers to exporting blob or clob type column content in binary format, which may have conflicts between column or row separators and field content. It supports a maximum data length of 64M for a single record, and errors will be reported if the length exceeds 64M;
- Base64: Refers to exporting blob or clob type column content using base64 encoding, which solves the problem of column or row separators conflicting with field content. It supports a maximum data length of 64M for a single record, and errors are reported when the length exceeds 64M;
- URL: Refers to exporting blob or clob type column content in a URL format, where each lob field content is saved as a separate file on disk. The exported main data file records the relative path of the lob file relative to the main data file.

Example

Example 1

```
./orato8a --encoding=base64
```

Example 2

```
./orato8a --encoding=url
```

Example 3

```
./orato8a -Ebase64
```

Example 4

```
./orato8a -Eurl
```

Parameter Description

Table -4111 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
encoding	E	text base64 url	text

4.11.2.2.4.18 help

function

This parameter is used to view help information for orato8a command parameters.

Example

```
--help
```

Parameter Description

Table -4112 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
help	nothing	nothing	nothing

4.11.2.2.4.19 version

function

This parameter is used to view the version information of the Orato8a tool.

Example

```
--version
```

Parameter Description

Table -4113 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
version	V	nothing	nothing

4.11.2.3 Example Description

To demonstrate, in Oracle's SSBM user, create a table lineorder with the following structure:

```
SQL> DESC lineorder;
Name                          Null?    Type
----------------------------- -------- ----------------
LO_ ORDERKEY                           NUMBER(18)
LO_ LINENUMBER                         NUMBER(18)
LO_ CUSTKEY                            NUMBER(18)
LO_ PARTKEY                            NUMBER(18)
LO_ SUPPERKEY                          NUMBER(18)
```

LO_ ORDERDATE	NUMBER(18)
LO_ ORDERPRIORITY	VARCHAR2(15)
LO_ SHIPPRIORITY	VARCHAR2(1)
LO_ QUANTITY	NUMBER(18)
LO_ EXTENDEDPRICE	NUMBER(18)
LO_ DISCOUNT	NUMBER(18)
LO_ REVENUE	NUMBER(18)
LO_ SUPPLYCOST	NUMBER(18)
LO_ TAX	NUMBER(18)
LO_ COMMIDATE	NUMBER(18)
LO_ SHIPMODE	VARCHAR2(10)

In this table, the test data for scale has already been loaded

SQL> SELECT COUNT(*) FROM lineorder;

COUNT(*)

---

six million one thousand two hundred and fifteen

4.11.2.3.1 Example of exporting full table data using parallel methods

Execute orato8a and set the table_ The name parameter sets the name of the table to be exported, and the parallel parameter can set the parallelism (<=CPU cores) to quickly export full table data.

$ ./ orato8a --user='ssbm/ ssbm@maya ' --table_ name=lineorder --file=/home/oracle/lineorder.txt --field=";" --format=3 --parallel=4

export columns: 17

export rows: 6001215

export time: 31 sec

process ok!

$ wc -l lineorder.txt

6001215 lineorder.txt

It can be seen that the exported file contains 6001215 rows of data.

4.11.2.3.2 Example of setting field delimiters

When exporting data, field delimiters can only be specified when the format parameter is specified as 3. This parameter has no default value. If this parameter is not specified during export, the exported data will not be surrounded by field delimiters. The examples in this section are based on the following table structure and data content.

```
DROP TABLE IF EXISTS message;
COMMIT;

CREATE TABLE    message(id int, name varchar2(20), message varchar2(50));

INSERT INTO message VALUES(1,'Tom','I am Tom');
Insert INTO message Values (2, 'Xiaoming', 'HHHHH "KKKK');
INSERT INTO message VALUES(3,'Peter','Hello!Hello!');
INSERT INTO message VALUES(4,'Yama','send mail');
INSERT INTO message VALUES(5,'Hellen','');
INSERT INTO message VALUES(6,'','');
INSERT INTO message VALUES(7,'Seven','a book store');
INSERT INTO message VALUES(8,'MMEE','yes');
INSERT INTO message VALUES(9,'George','Thank you.');
INSERT INTO message VALUES(10,'Lastman','no message');
COMMIT;
```

4.11.2.3.2.1 Set visible characters as field delimiters

In this example, single quotation marks (') are used as field delimiters, and the export method is as follows:

```
$ ./ orato8a --user= 'ssbm/ ssbm@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field='|@|' --string_ qualifier="'"
export columns: 3
export rows: 10
export time: 0 sec
process ok!
$ cat message.tbl
'1'|@|'Tom'|@|'I am Tom'
Xiaoming
'3'|@|'Peter'|@|'Hello! Hello!'
'4'|@|'Yama'|@|'send mail'
```

```
'5'|@|'Hellen'|@|
'6'|@||@|
'7'|@|'Seven'|@|'a book store'
'8'|@|'MMEE'|@|'yes'
'9'|@|'George'|@|'Thank you.'
'10'|@|'Lastman'|@|'no message'
```

4.11.2.3.2.2 Set field enclosing character with escape character

In this example, the horizontal tab is used as the field enclosing character. The horizontal tab is specified by the escape character " t". The export method is as follows:

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field='|@|' --string_ qualifier="\t"
export columns: 3
export rows: 10
export time: 0 sec
process ok!
$ cat message.tbl
      1          |@|        Tom        |@|        I am Tom
      2 | @ | Xiaoming | @ | HHHH "KKKK
      3          |@|        Peter     |@|        Hello! Hello!
      4          |@|        Yama      |@|        send mail
      5          |@|        Hellen    |@|
      6          |@||@|
      7          |@|        Seven     |@|        a book store
      8          |@|        MMEE      |@|        yes
      9          |@|        George    |@|        Thank you.
     10         |@|        Lastman |@|        no message
```

4.11.2.3.2.3 Setting Field Enclosers Using Hexadecimal Method

- The hexadecimal method specifies a wide range of export parameters, which can specify both visible and invisible characters.
- This example provides two methods of using hexadecimal to represent field bounding box parameters. Both methods set the plus sign (+) as the field bounding box and use hexadecimal to represent " x2b" or "x'2b '". The export method is as follows:

Example 1

Set the bounding box using the hexadecimal method ' x2b'.

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field='|@|' --string_ qualifier="'\x2b"
export columns: 3
export rows: 10
export time: 0 sec
process ok!
$ cat message.tbl
+1+|@|+Tom+|@|+I am Tom+
+2+| @ |+Xiaoming+| @ |+HHHHH "KKKK+
+3+|@|+Peter+|@|+Hello! Hello!+
+4+|@|+Yama+|@|+send mail+
+5+|@|+Hellen+|@|
+6+|@||@|
+7+|@|+Seven+|@|+a book store+
+8+|@|+MMEE+|@|+yes+
+9+|@|+George+|@|+Thank you.+
+10+|@|+Lastman+|@|+no message+
```

Example 2

Use the hexadecimal method "x'2b '" to set the bounding box.

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field='|@|' --string_ qualifier="'x'2b'"
export columns: 3
export rows: 10
export time: 0 sec
process ok!
$ cat message.tbl
+1+|@|+Tom+|@|+I am Tom+
+2+| @ |+Xiaoming+| @ |+HHHHH "KKKK+
+3+|@|+Peter+|@|+Hello! Hello!+
+4+|@|+Yama+|@|+send mail+
+5+|@|+Hellen+|@|
+6+|@||@|
```

```
+7+|@|+Seven+|@|+a book store+
+8+|@|+MMEE+|@|+yes+
+9+|@|+George+|@|+Thank you.+
+10+|@|+Lastman+|@|+no message+
```

4.11.2.3.2.4 Export data contains field delimiters

If the exported data contains field delimiters, Orato8a will use field delimiters to escape the data content. The export method is as follows:

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field='|@|' --string_ qualifier="\""
export columns: 3
export rows: 10
export time: 0 sec
process ok!
$ cat message.tbl
"1"|@|"Tom"|@|"I am Tom"
2 | @ | Xiaoming | @ | HHHH KKKK
"3"|@|"Peter"|@|"Hello!Hello!"
"4"|@|"Yama"|@|"send mail"
"5"|@|"Hellen"|@|
"6"|@||@|
"7"|@|"Seven"|@|"a book store"
"8"|@|"MMEE"|@|"yes"
"9"|@|"George"|@|"Thank you."
"10"|@|"Lastman"|@|"no message"
```

Upon reviewing the exported data content, it was found that the last column of the second row of data originally contained 'HHHH "KKKK'. However, due to the double quotation marks (") contained in it being the same as the field delimiter, they were escaped using the field delimiter.

4.11.2.3.3 Example of Setting Row Delimiters

When exporting data, the row separator parameter can only be specified when the format parameter is 3. If this parameter is not specified, the default row separator for exported

data is' n '. The examples in this section are based on the following table structure and data content.

be careful

The line separator cannot be specified as the same content as the data content or other control character, otherwise it may cause disambiguation

Meaning, causing data to not be loaded back into the table.

```
DROP TABLE IF EXISTS message;
COMMIT;

CREATE TABLE    message(id int, name varchar2(20), message varchar2(50));

INSERT INTO message VALUES(1,'Tom','I am Tom');
Insert INTO message Values (2, 'Xiaoming', 'HHHHH "KKKK');
INSERT INTO message VALUES(3,'Peter','Hello!Hello!');
INSERT INTO message VALUES(4,'Yama','send mail');
INSERT INTO message VALUES(5,'Hellen','');
INSERT INTO message VALUES(6,'','');
INSERT INTO message VALUES(7,'Seven','a book store');
INSERT INTO message VALUES(8,'MMEE','yes');
INSERT INTO message VALUES(9,'George','Thank you.');
INSERT INTO message VALUES(10,'Lastman','no message');
COMMIT;
```

4.11.2.3.3.1 Set visible characters as line separators

In this example, the underscore (_) is used as the row separator, and the export method is as follows:

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --query="select id,name from message where
rownum<=5;" --file='./ message.tbl' --format=3 --field='|' --string_ qualifier="'" --line_
separator='_'
export columns: 2
export rows: 5
export time: 0 sec
process ok!
$ cat message.tbl
'1'|'Tom'_' Xiao Ming 3'|'Peter'_' 4'|'Yama'_' 5'|'Hellen'_
```

4.11.2.3.3.2 Set line delimiter with escape character

In this example, an escape character is used to specify an invisible character as the line separator. This invisible character represents the system ringtone, and the escape character represents' a '. The export method is as follows:

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --query="select id,name from message where rownum>=5;" --file='./ message.tbl' --format=3 --field='|' --string_ qualifier="'" --line_ separator='\a'
export columns: 2
export rows: 5
export time: 0 sec
process ok!
$ cat message.tbl
1 '| Tom' 2 '| Xiao Ming' 3 '| Peter' 4 '| Yama' 5 '| Hellen'
```

Due to being an invisible character, it cannot be viewed directly using the cat command. Using the - e parameter to view the result is as follows:

```
$ cat –e message.tbl
'1'|'Tom'^G'2'|'M-eM-0M-^OM-fM-^XM-^N'^G'3'|'Peter'^G'4'|'Yama'^G'5'|'Hellen'^G
```

explain

The part represented by '^ G' is the line separator ' a' we specified.

4.11.2.3.3.3 Setting Row Delimiters Using Hexadecimal Method

In this example, there are two methods of using hexadecimal to represent row separators. Both methods use ' n' as the row separator and use hexadecimal to represent ' x0a' or 'x'0a'. The export method is as follows:

Example 1

Set the line separator using the hexadecimal method ' x0a'.

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --query="select * from message;" --file='./ message.tbl' --format=3 --field='|' --line_ separator="\x0a"
export columns: 3
export rows: 10
export time: 0 sec
process ok!
$ cat message.tbl
```

```
1|Tom|I am Tom
2 | Xiaoming | HHHH "KKKK
3|Peter|Hello! Hello!
4|Yama|send mail
5|Hellen|
6||
7|Seven|a book store
8|MMEE|yes
9|George|Thank you.
10|Lastman|no message
```

Example 2

Set the line separator using the hexadecimal method 'x'0a'.

```
--file='./ message.tbl' --format=3 --field='|' --line_ separator="x'0a'"
export columns: 3
export rows: 10
export time: 0 sec
process ok!
$ cat message.tbl
1|Tom|I am Tom
2 | Xiaoming | HHHH "KKKK
3|Peter|Hello! Hello!
4|Yama|send mail
5|Hellen|
6||
7|Seven|a book store
8|MMEE|yes
9|George|Thank you.
10|Lastman|no message
```

4.11.2.3.4 Example of setting a NULL value

When exporting data, a NULL value can only be specified when the format parameter is specified as 3. If this parameter is not specified during export, the NULL value in the

data will be exported as an empty string. It should be noted that the examples in this section are based on the following table structure and data content.

 be careful

The NULL value cannot be specified as the same content as the data content or other control character, otherwise it may cause ambiguity, and the data cannot be loaded back into the table.

```
DROP TABLE IF EXISTS message;
COMMIT;

CREATE TABLE    message(id int, name varchar2(20), message varchar2(50));

INSERT INTO message VALUES(1,'Tom','I am Tom');
Insert INTO message Values (2, 'Xiaoming', 'HHHHH "KKKK');
INSERT INTO message VALUES(3,'Peter','Hello!Hello!');
INSERT INTO message VALUES(4,'Yama','send mail');
INSERT INTO message VALUES(5,'Hellen','');
INSERT INTO message VALUES(6,'','');
INSERT INTO message VALUES(7,'Seven','a book store');
INSERT INTO message VALUES(8,'MMEE','yes');
INSERT INTO message VALUES(9,'George','Thank you.');
INSERT INTO message VALUES(10,'Lastman','no message');
COMMIT;
```

4.11.2.3.4.1 Set visible strings to represent NULL values

In this example, the NULL value in the data is represented as'% null% ', and the export method is as follows:

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field=';' -- null_ value='%null%'
export columns: 3
export rows: 10
export time: 0 sec
process ok!
$ cat message.tbl
1; Tom; I am Tom
```

```
2; Xiaoming; HHHH"KKKK
3; Peter; Hello! Hello!
4; Yama; send mail
5; Hellen;% null%
6;% null%;% null%
7; Seven; a book store
8; MMEE; yes
9; George; Thank you.
10; Lastman; no message
```

4.11.2.3.4.2 Set NULL value with escape character

In this example, set to export the NULL values in the data as' tnull '(i.e. a horizontal tab, 4 letter nulls, and a backslash). The export method is as follows:

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field=';' -- null_ value='\tnull\\'
export columns: 3
export rows: 10
export time: 0 sec
process ok!
$ cat message.tbl
1; Tom; I am Tom
2; Xiaoming; HHHH"KKKK
3; Peter; Hello! Hello!
4; Yama; send mail
5; Hellen;              null\
6;          null\;    null\
7; Seven; a book store
8; MMEE; yes
9; George; Thank you.
10; Lastman; no message
```

4.11.2.3.4.3 Setting NULL values using hexadecimal notation

In this example, two methods of using hexadecimal to represent NULL values are provided, both of which use a vertical tab and a system bell note to represent NULL

values. The hexadecimal method is represented as " x0b x07" or "x'0b07 '", and the export method is as follows:

Example 1

Set NULL value using hexadecimal method ' x0b x07'

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field=';' -- null_ value="\x0b\x07"
export columns: 3
export rows: 10
export time: 0 sec
process ok!
```

Due to the fact that the exported data file contains invisible characters, we use the - e parameter of the cat command to display the invisible characters in the file. The vertical tab is displayed as ^ K in this way, and the system ringtone is displayed as ^ G:

```
$ cat -e message.tbl
1; Tom; I am Tom$
2; M-eM-0M-^OM-fM-^XM-^N; HHHH"KKKK$
3; Peter; Hello! Hello!$
4; Yama; send mail$
5; Hellen;^ K^G$
6;^ K^G;^ K^G$
7; Seven; a book store$
8; MMEE; yes$
9; George; Thank you.$
```

Example 2

Use the hexadecimal method "x'0b07 '" to set a NULL value.

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field=';' -- null_ value="x'0b07'"
export columns: 3
export rows: 10
export time: 0 sec
process ok!
```

```
$ cat -e message.tbl
1; Tom; I am Tom$
2; M-eM-0M-^OM-fM-^XM-^N; HHHH"KKKK$
3; Peter; Hello! Hello!$
4; Yama; send mail$
5; Hellen;^ K^G$
6;^ K^G;^ K^G$
7; Seven; a book store$
8; MMEE; yes$
9; George; Thank you.$
10; Lastman; no message$
```

4.11.2.3.5 Export table data for non logged in users

- When exporting data, the content specified by the -- user parameter is called the login user. When exporting data, the user name specified by this parameter and other related connection parameters are used to log in to the Oracle database for data export;
- If you want to export data for tables created by non logged in users, you also need to use -- owner_ The name parameter specifies the username of the user who created the table;
- Assuming we want to export the table message created by the ssbm user using the expdata user, first we need to grant the expdata user access to the dba_ Extents and dba_ The SELECT permission for the objects table is required for all login users exporting data to Orato8a to have SELECT permission for these two tables, and then the expdata user needs to be granted SELECT permission for the message table.

Step 1

Authorization method, first create a regular user expdata as a system administrator:

```
SQL> conn /as sysdba;
Connected.
SQL> create user expdata identified by hello;

User created.

SQL> alter user expdata account unlock;

User altered.
```

```
SQL> grant create session to expdata;

Grant succeeded.
```

Step 2

Then, still using the system administrator identity, grant expdata access to the dba_ Extents and dba_ Select permission for the objects table:

```
SQL> grant select on dba_ extents to expdata;

Grant succeeded.

SQL> grant select on dba_ objects to expdata;

Grant succeeded.
```

Step 3

Finally, the ssbm user should be used to grant select permission to the message table to the expdata user:

```
SQL> conn ssbm/ssbm;
Connected.
SQL> grant select on message to expdata;
```

Step 4

Export using:

```
$ ./ orato8a --user='expdata/ hello@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field='@' --owner_ name="ssbm"
export columns: 3
export rows: 10
export time: 0 sec
process ok!
$ cat message.tbl
1@ Tom@I    am Tom
2 @ Xiaoming @ HHHH    "KKKK
3@ Peter@Hello ! Hello!
4@ Yama@send    mail
5@Hellen@\N
6@\N@\N
```

```
7@ Seven@a    book store
8@ MMEE@yes
9@ George@Thank    you.
10@Lastman @no message
```

4.11.2.3.6 Export table data with clob or blob type fields

- In this example, a method is provided to export clob or blob field data as an unfixed length text file.
- By specifying - blob_ The conf parameter is used to store clob or blob field data greater than or equal to 32kB on HBase or HDFS, while clob or blob field data less than 32kB is stored directly in a text file.
- For the storage location of clob or blob field data, please refer to - blob_ Description of the conf parameter.
- Due to the possibility of column delimiters or line breaks in clob or blob fields, the use of bounding boxes is necessary.

The export method is as follows:

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field=';' -- blob_ conf=orato8a.xml --string_ qualifier="\x2b"
export columns: 3
export rows: 10
export time: 0 sec
process ok!
```

explain

By specifying the -- encoding parameter, set the clob or blob field data to be stored as text (binary), base64 (base64 encoding), or URL (external file).

```
$ ./ orato8a --user='ssbm/ ssbm@maya ' --table_ name='message' --file='./ message.tbl'
--format=3 --field=';' -- encoding=base64 --string_ qualifier="\x2b"
export columns: 3
export rows: 10
export time: 0 sec
process ok!
```

 **be careful**

--blob_ The conf parameter has a higher priority than -- encoding, using blob_ When in conf mode, clob or

Blob data is stored in binary format in text files, HBase, or HDFS.

4.11.3 Use of db2to8a tool

Db2to8a is a standalone data extraction tool that needs to be deployed on clients that can access db2s, or directly on a service with db2server.

4.11.3.1 Installation File Description

The db2to8a installation package is provided in a compressed form of tar. bz2. For example: db2to8a_ 24816_ Redhat6.2_

x86_ 64.tar.bz2。

explain

In the tar.bz2 file

- Db2to8a: represents the name of the tool;
- 24816: represents the version number of the tool;
- Redhat 6.2: represents that the operating system running this tool is Red hat 6.2;
- x86_ 64: indicates that the tool is a tool running on a 64 bit operating system.

4.11.3.2 Obtain installation files and unzip for installation

We take the 64 bit Linux operating system as an example to introduce to users how to extract and install db2to8a after obtaining it.

Step 1

Insert the installation CD into the CD drive and use the mount command to mount the CD drive to the Linux file system. The command to load the optical drive is as follows (usually, root is required to load the optical drive):

```
# mkdir /mnt/cdrom
```

```
# mount /dev/cdrom /mnt/cdrom
```

Step 2

Copy the compressed file of the installation package in the optical drive (such as db2to8a_24816_Redhat6.2_x86_64. tar. bz2) to a directory on the file system. Enter the directory (assuming the installation package is placed in the/root directory) and use the tar command in command line mode to decompress. The decompression command is as follows:

```
# tar xfj db2to8a_ 24816_ Redhat6.2_ x86_ 64.tar.bz2
NM_ CONTROLLED=no
```

Step 3

After decompression, an executable db2to8a program will be directly generated in the decompression directory.

```
$ ll
Total usage 2068
……
-Rw-r - r -1 root 1380535 August 23 01:08 db2to8a
-Rw-r - r -1 root 663929 August 22nd 17:13 db2to8a_ 24816_ Redhat6.2_ x86_ 64.tar.bz2
```

4.11.3.3 Grammar format

```
./db2to8a parameter_ 1 parameter_ 2 ... parameter_ n
```

explain

- parameter_ 1: The parameters of db2to8a can be used multiple times after a db2to8a. The parameters can be in full name format or abbreviated format;
- Full name format: -- parameter_ 1=parameter value,=no spaces on either side;
- Abbreviation format: - parameter_ When abbreviating parameter values, there cannot be spaces between the parameters and parameter values.

be careful

The user executing db2to8a must be a user who can access the database.

Example

Below is a simple example to illustrate how the db2to8a tool extracts data from DB2. In this example, first log in to DB2, and then write a query SQL that queries a piece of data in table t in the test library; Exit DB2 and use the db2to8a tool to extract query SQL result data to verify the functionality of db2to8a in extracting data.

```
$ db2
(c) Copyright IBM Corporation 1993,2007
Command Line Processor for DB2 Client 9.7.1

You can issue database manager commands and SQL statements from the command
prompt. For example:
    db2 => connect to sample
    db2 => bind sample.bnd

For general help, type: ?.
For command help, type: ? command, where command can be
the first few keywords of a database manager command. For example:
 ? CATALOG DATABASE for help on the CATALOG DATABASE command
 ? CATALOG              for help on all of the CATALOG commands.

To exit db2 interactive mode, type QUIT at the command prompt. Outside
interactive mode, all commands must be prefixed with 'db2'.
To list the current command option settings, type LIST COMMAND OPTIONS.

For more detailed help, refer to the Online Reference Manual.

db2 => connect to test

   Database Connection Information

 Database server        = DB2/LINUXX8664 9.7.1
 SQL authorization ID   = DB2INST1
 Local database alias   = TEST
```

```
db2 => SELECT * FROM T fetch first 1 rows only

A           B              C                         D                          E
------ -------- --------------- ------------------- ----------
     -3         0.93 helloworld                    +5.69900E+002 03/19/2013
1 record(s) selected.
```

Then exit DB2 and use db2to8a to extract the data queried above.

```
$ cd test
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select * from t fetch first 1 rows only"
-f'data1.txt' -m'0' -e'|'
you machine is Little endian!

   Connecting to test...
   Connected to test.

--- unload [text file] mode ---
--- field="|" ---
            0 rows exported at 2013-08-31 14:18:24
            1 rows exported at 2013-08-31 14:18:24
output file data1.txt closed
  export:                1 rows.
  export:                5 columns.
  export time:        0.01 sec

   Disconnecting from test...
   Disconnected from test.

$ cat data1.txt
-3|0.93|helloworld|5.699000E+02|2013-03-19
```

The data in the exported data source file is consistent with the data queried in th e DB2 system.

4.11.3.4 Parameter Description

4.11.3.4.1 db_ name

function

Specify the database name.

Example

Example 1

```
--db_ name='test'
```

Example 2

```
-D'test'
```

Parameter Description

Table -4114 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
db_ name	D	nothing	nothing

4.11.3.4.2 user

function

Specify the user name to access the DB2 database.

Example

Example 1

```
--user='db2user'
```

Example 2

```
-u'db2user'
```

Parameter Description

Table -4115 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
user	u	nothing	nothing

4.11.3.4.3password

function

Specify the user password for accessing the database of the database.

Example

Example 1

```
--password='db2user'
```

Example 2

```
-p'db2user'
```

explain

Table -4116 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
password	p	nothing	nothing

4.11.3.4.4query

function

- Specify the query SQL statement used for exporting data. Due to the frequent occurrence of spaces in query statements, this parameter refers to

  Timing requires double quotation marks to limit;

- The query statement should be an SQL that conforms to the DB2 syntax and only returns a set of results.

Example

Example 1

```
--query="select * from lineorder"
```

Example 2

```
-q"select * from lineorder"
```

Parameter Description

Table -4117 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
query	q	nothing	nothing

4.11.3.4.5file

function

This parameter specifies the name of the exported data file.

explain

This parameter can specify a file name that contains an absolute path or a file name that contains a relative path. When

When this parameter specifies a file name that does not contain a path, the file is saved in the current path.

Example

Example 1

```
--file='aaa.txt'

--file='/home/lina/aaa.txt'
```

Example 2

```
-f'aaa.txt'

-f'/home/lina/aaa.txt'
```

Parameter Description

Table -4118 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
file	f	nothing	nothing

4.11.3.4.6format

function

This parameter controls the data format of the exported data file.

explain

When set to 3, the exported data is in text format without escape.

Example

Example 1

```
--format='3'
```

Example 2

```
-m'3'
```

Parameter Description

Table -4119 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
format	m	three	three

4.11.3.4.7string_ qualifier

function

This parameter is used to set the field bounding box, and is only valid when for mat=3. If a field bounding box is set, all fields will be added with a field bound ing box.

Example

Field delimiters can be specified using hexadecimal.

Example 1

```
--string_ qualifier ='@'
```

Example 2

```
-s'@'
```

If the field delimiter is set to hexadecimal, the command parameter is followed by double quotes surrounding the hexadecimal value.

Example 3

```
--string_ qualifier ="x'62'"
```

Example 4

```
-s"x'62'"
```

Parameter Description

Table 4120 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
string_ qualifier	s	One character	nothing

4.11.3.4.8 field

function

This parameter is used to set the field separator. The text format can use hexadecimal.

explain

When using format=3, the value of this parameter must be set.

Example

Example 1

```
--field ="\x2c"
```

Example 2

```
-e"\x2c"
```

Parameter Description

Table -4121 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
field	e	15 characters	nothing

4.11.3.4.9line_ separator

function

This parameter is used to set the row separator. This parameter is only valid whe n format=3.

Example

Example 1

```
--line_ separator ='@'
```

Example 2

```
-l'@'
```

If set to a hexadecimal line separator, the command parameter is followed by do uble quotes surrounding the hexadecimal value.

Example 1

```
--line_ separator ="x'62'"
```

Example 2

```
-l"x'62'"
```

Parameter Description

Table -4122 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
line_ separator	l	Single character	'\n'

 **be careful**

Only single byte line breaks are supported in format=5 loading mode.

4.11.3.4.10 null_ value

function

This parameter is used to set a NULL value.

 **explain**

When format=3, this parameter is valid, but there is no default value.

Example

Example 1

```
--null_ value ='\N'
```

Example 2

```
-n'\N'
```

Parameter Description

Table -4123 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
null_ value	n	Maximum 15 characters	Format=3: None

4.11.3.4.11 help

function

This parameter is used to view help information for the db2to8a command param eters.

Example

```
--help
```

Parameter Description

Table -4124 Parameter Description

Full name of parameter	Parameter abbreviation	Value range	Default value
help	nothing		

4.11.3.5 Example Description

4.11.3.5.1 Example of setting field delimiters

Create the following tables and data in the DB2 database system:

```
CREATE DATABASE test
connect to test
drop table t
CREATE TABLE t (a int,b decimal(15,2), c varchar(20),d real,e date)

INSERT INTO t VALUES (-3,0.93,'helloworld',569.9,'2013-03-19')
INSERT INTO t VALUES (-3,0.93,'helloworld',569.9,'2013-03-19')
INSERT INTO t VALUES (19,-15.69,'ff',157,'1999-11-16')
INSERT INTO t VALUES (20,16,'he\n%\n\t\t$\n',52,'1982-06-03')
INSERT INTO t VALUES (20,16,'\n\nhe\t%$\t',23,'1982-06-03')
INSERT INTO t VALUES (58,12.30,'xyz       abc',13.69,'1978-12-20')
Insert INTO t Values (991, -0, 'Beijing', 1.230000E+01, '1989 06 07')
```

4.11.3.5.1.1 Exporting data with delimiters

In this example, the exported data contains the set bounding box and the boundin g box is set to visible characters.

```
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select *    from t" -f'data1.txt' -m'3' -e'|' -l'\n'
-s'h'
you machine is Little endian!


   Connecting to test...
   Connected to test.


--- unload [text file] mode ---
--- field="|" ---
```

```
            0 rows exported at 2013-08-30 13:33:29
            7 rows exported at 2013-08-30 13:33:29
output file data1.txt closed
 export:                7 rows.
 export:                5 columns.
 export time:       0.00 sec


   Disconnecting from test...
   Disconnected from test.


$ cat data1.txt
h-3h|h0.93h|hhhelloworldh|h5.699000E+02h|h2013-03-19h
h-3h|h0.93h|hhhelloworldh|h5.699000E+02h|h2013-03-19h
h19h|h-15.69h|hffh|h1.570000E+02h|h1999-11-16h
h20h|h16.00h|hhhe\n%\n\t\t$\nh|h5.200000E+01h|h1982-06-03h
h20h|h16.00h|h\n\nhhe\t%$\th|h2.300000E+01h|h1982-06-03h
h58h|h12.30h|hxyz        abch|h1.369000E+01h|h1978-12-20h
H991h | h0.00h | h Beijing h | h1.230000E+01h | h1989-06-07h
```

4.11.3.5.1.2 Set as a bounding box for invisible characters

In this example, the exported data contains the set bounding box, and the boundi ng box is set to invisible characters.

```
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select *    from t" -f'data1.txt' -m'3' -e'|' -l'\n'
-s'\x09'
you machine is Little endian!


   Connecting to test...
   Connected to test.


--- unload [text file] mode ---
--- field="|" ---
            0 rows exported at 2013-08-30 13:34:18
            7 rows exported at 2013-08-30 13:34:18
output file data1.txt closed
 export:                7 rows.
```

```
export:              5 columns.
export time:     0.00 sec


  Disconnecting from test...
  Disconnected from test.
```

$ cat data1.txt

Due to the large number of data columns included, the exported results in this example are displayed using segmented screenshots.

Firstly, display the export results of the first two columns

```
-3      |       0.93    |
-3      |       0.93    |
19      |       -15.69  |
20      |       16.00   |
20      |       16.00   |
58      |       12.30   |
991     |       0.00    |
```

Finally, display the export results of the remaining columns

```
hel l owor l d      |       5. 699000E+02   |       2013-03-19
hel l owor l d      |       5. 699000E+02   |       2013-03-19
f f         |       1. 570000E+02   |       1999-11-16
he\n%\n\t \t $\n    |       5. 200000E+01   |       1982-06-03
\n\nhe\t %$\t       |       2. 300000E+01   |       1982-06-03
xyz                 abc     |       1. 369000E+01   |       1978-12-20
北京    |       1. 230000E+01   |       1989-06-07
```

explain

\X09 represents the function of tab tab, and in the exported data, the string "xyz abc" contains an interval between xyz and abc for a tab tab.

4.11.3.5.1.3 The exported data does not contain the set bounding box (1)

In this example, the exported data does not contain the set bounding box, and th e bounding box is set to visible characters.

$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select * from t" -f'data1.txt' -m'3' -e'|' -l'\n' -s'""'

```
you machine is Little endian!


  Connecting to test...
```

```
   Connected to test.


--- unload [text file] mode ---
--- field="|" ---
            0 rows exported at 2013-08-30 13:33:51
            7 rows exported at 2013-08-30 13:33:51
output file data1.txt closed
 export:                7 rows.
 export:                5 columns.
 export time:       0.00 sec


   Disconnecting from test...
   Disconnected from test.


$ cat data1.txt
"-3"|"9.30000000000000E-001"|"helloworld"|"569.9"|"12:15:19"
"-3"|"9.30000000000000E-001"|"helloworld"|"569.9"|"12:15:19"
"19"|"-1.56900000000000E+001"|"^f^f^"|" a  b  c abc"|"22:17:26"
"20"|"1.60000000000000E+001"|"he\n%\n\t\t$\n"|"dasfal ""/\^5"|"09:15:21"
"20"|"1.60000000000000E+001"|"\n\nhe\t%$\t"|"dasdf    dasf;l    "|"17:16:29"
"58"|"1.23000000000000E+001"|"xyz              abc"|"erEERE"|"00:00:00"
991 | "0.000 000000 E+000" | "North ^ ^ ffd # $Jingtian dt Tianjin" | "0123" | "23:59:59"
||"null"|"null"|"17:16:29"
|||"null"|"22:17:26"
```

4.11.3.5.1.4 The exported data does not contain the set bounding box (2)

In this example, the exported data does not contain the set bounding box, and th e bounding box is set to hexadecimal characters.

```
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select *    from t" -f'data1.txt' -m'3' -e'|' -l'\n'
-s'\x61'
you machine is Little endian!


   Connecting to test...
   Connected to test.
```

```
--- unload [text file] mode ---
--- field="|" ---
            0 rows exported at 2013-08-30 13:34:18
            7 rows exported at 2013-08-30 13:34:18
output file data1.txt closed
 export:                7 rows.
 export:                5 columns.
 export time:       0.00 sec

   Disconnecting from test...
   Disconnected from test.

$ cat data1.txt
a-3a|a0.93a|ahelloworlda|a5.699000E+02a|a2013-03-19a
a-3a|a0.93a|ahelloworlda|a5.699000E+02a|a2013-03-19a
a19a|a-15.69a|affa|a1.570000E+02a|a1999-11-16a
a20a|a16.00a|ahe\n%\n\t\t$\na|a5.200000E+01a|a1982-06-03a
a20a|a16.00a|a\n\nhe\t%$\ta|a2.300000E+01a|a1982-06-03a
a58a|a12.30a|axyz       aabca|a1.369000E+01a|a1978-12-20a
A991a | a0.00a | a Beijing a | a1.230000E+01a | a1989-06-07a
```

4.11.3.5.2 Example of Setting Row Delimiters

Create the following tables and data in the DB2 database system:

```
CREATE DATABASE test
connect to test
DROP TABLE t
CREATE TABLE t (a int,b decimal(15,2), c varchar(20),d real,e date)

INSERT INTO t VALUES (-3,0.93,'helloworld',569.9,'2013-03-19')
INSERT INTO t VALUES (-3,0.93,'helloworld',569.9,'2013-03-19')
INSERT INTO t VALUES (19,-15.69,'ff',157,'1999-11-16')
INSERT INTO t VALUES (20,16,'he\n%\n\t\t$\n',52,'1982-06-03')
INSERT INTO t VALUES (20,16,'\n\nhe\t%$\t',23,'1982-06-03')
INSERT INTO t VALUES (58,12.30,'xyz       abc',13.69,'1978-12-20')
Insert INTO t Values (991, -0, 'Beijing', 1.230000E+01, '1989 06 07')
```

4.11.3.5.2.1 Export data does not contain row delimiters (1)

In this example, the exported data does not contain the set row delimiter, and the row delimiter is set to visible characters.

```
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select *    from t" -f'data1.txt' -m'3' -e'|' -l's'
you machine is Little endian!


   Connecting to test...
   Connected to test.


--- unload [text file] mode ---
--- field="|" ---
            0 rows exported at 2013-08-30 13:35:47
            7 rows exported at 2013-08-30 13:35:47
output file data1.txt closed
 export:              7 rows.
 export:              5 columns.
 export time:      0.00 sec


   Disconnecting from test...
   Disconnected from test.


$ cat data1.txt
            -3 | 0.93 | helloworld | 5.699000E+02 | 2013-03-19s-3 | 0.93 | helloworld | 5.69900E+02 |
            2013-03-19s19 | -15.69 | ff | 1.570000E+02 | 1999-11-16s20 | 16.00 | he    n%    n    t    t $ n
            | 5.200000E+01 | 1982-06-03s20 | 16.00 |    n    nhe    t $ t | 2.300000E+01 | 1982-06-03s58 |
            12.30 | xyz abc | 1.369000E+01 | 1978-12-20s991 | 0.00 | Beijing | 1.230000 E+01 |
            1989-06-07s
```

4.11.3.5.2.2 Export data does not contain row delimiters (2)

In this example, the exported data does not contain the set row delimiter, and the row delimiter is set to an invisible character.

```
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select *    from t" -f'data1.txt' -m'3' -e'|' -l'\n'
you machine is Little endian!
```

```
   Connecting to test...
   Connected to test.


--- unload [oracle text file] mode ---
--- field="|" ---
--- record="
" ---
            0 rows exported at 2013-10-18 17:19:13
            7 rows exported at 2013-10-18 17:19:13
output file ./ wzx/data1.txt closed
 export:                7 rows.
 export:                5 columns.
 export time:       0.00 sec


   Disconnecting from test...
   Disconnected from test.


$ cat data1.txt
-3|0.93|helloworld|5.699000E+02|2013-03-19
-3|0.93|helloworld|5.699000E+02|2013-03-19
19|-15.69|ff|1.570000E+02|1999-11-16
20|16.00|he\n%\n\t\t$\n|5.200000E+01|1982-06-03
20|16.00|\n\nhe\t%$\t|2.300000E+01|1982-06-03
58|12.30|xyz       abc|1.369000E+01|1978-12-20
991 | 0.00 | Beijing | 1.230000E+01 | 1989-06-07
```

4.11.3.5.2.3 Export data does not contain row delimiters (3)

In this example, the exported data does not contain the set row delimiter, and the row delimiter is set to hexadecimal characters.

```
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select *    from t" -f'data1.txt' -m'3' -e'|'
-l"x'62'"
you machine is Little endian!


   Connecting to test...
   Connected to test.
```

```
--- unload [oracle text file] mode ---
--- field="|" ---
--- record="x'62'" ---
            0 rows exported at 2013-10-18 17:21:49
            7 rows exported at 2013-10-18 17:21:49
output file ./ wzx/data1.txt closed
 export:                7 rows.
 export:                5 columns.
 export time:       0.00 sec

   Disconnecting from test...
   Disconnected from test.

$ cat data1.txt
            -3 | 0.93 | helloworld | 5.699000E+02 | 2013-03-19b-3 | 0.93 | helloworld | 5.69900E+02 |
            2013-03-19b19 | -15.69 | ff | 1.570000E+02 | 1999-11-16b20 | 16.00 | he    n%    n    t    t $ n
            | 5.200000E+01 | 1982-06-03b20 | 16.00 |    n    nhe    t $ t | 2.300000E+01 | 1982-06-03b58
            | 12.30 | xyz abc | 1.369000E+01 | 1978-12-20b991 | 0.00 | Beijing | 1.230000 E+01 |
            1989-06-07b
```

4.11.3.5.3 Example of setting field delimiters

Create the following tables and data in the DB2 database system:

```
CREATE DATABASE test
connect to test
DROP TABLE t1
CREATE TABLE t1(i int,j bigint,c char(20),v varchar(255))
INSERT INTO t1 VALUES(12,34,'ajhsa','7shuusa8us')
INSERT INTO t1(i) VALUES(89)
INSERT INTO t1 VALUES(0,90,'sniuda','djkjkdsd')
INSERT INTO t1 VALUES(109,0,' ',' ')
INSERT INTO t1 VALUES(850266,9055655988595,'iu92uijw9i218uiw9w','wjijs')
INSERT INTO t1(i,c,v) VALUES(-19982921,'s','abc')
```

4.11.3.5.3.1 Set Field Delimiter (1)

In this example, the exported data file is in plain text format, and the field separator is set to '|'.

```
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select * from t1"    -m'0' -e'|'    -f'data1.txt'
you machine is Little endian!


   Connecting to test...
   Connected to test.


--- unload [text file] mode ---
--- field="|" ---
            0 rows exported at 2013-08-30 13:40:47
            6 rows exported at 2013-08-30 13:40:47
output file t13.dat10 closed
 export:                6 rows.
 export:                4 columns.
 export time:       0.00 sec


   Disconnecting from test...
   Disconnected from test.


$ cat data1.txt
12|34|ajhsa                        |7shuusa8us
89|\N|\N|\N
0|90|sniuda                       |djkjkdsd
109|0|                             |
850266|9055655988595|iu92uijw9i218uiw9w    |wjijs
-19982921|\N|s                              |abc
```

4.11.3.5.3.2 Set Field Delimiter (2)

In this example, the exported data file is in Oracle text format, and the field separator is set to ';'.

```
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select * from t1"    -m'3' -e';' - s'"' -n'NULL'
-f'data1.txt'
```

```
you machine is Little endian!


   Connecting to test...
   Connected to test.


--- unload [text file] mode ---
--- field=";" ---
            0 rows exported at 2013-08-30 13:41:14
            6 rows exported at 2013-08-30 13:41:14
output file t13.dat11 closed
  export:                6 rows.
  export:                4 columns.
  export time:       0.00 sec


   Disconnecting from test...
   Disconnected from test.


$ cat data1.txt
"12"; "34"; "ajhsa                          "; "7shuusa8us"
"89"; "NULL"; "NULL"; "NULL"
"0"; "90"; "sniuda                        "; "djkjkdsd"
"109"; "0"; "                                 "; " "
"850266"; "9055655988595"; "iu92uijw9i218uiw9w    "; "wjijs"
"-19982921"; "NULL"; "s                                  "; "abc"
```

4.11.3.5.4 Example of exporting data with null values

Create the following tables and data in the DB2 database system:

```
CREATE DATABASE test
connect to test
DROP TABLE t1
CREATE TABLE t1(i int,j bigint,c char(20),v varchar(255))
INSERT INTO t1 VALUES(12,34,'ajhsa','7shuusa8us')
INSERT INTO t1(i) VALUES(89)
INSERT INTO t1 VALUES(0,90,'sniuda','djkjkdsd')
INSERT INTO t1 VALUES(109,0,' ',' ')
INSERT INTO t1 VALUES(850266,9055655988595,'iu92uijw9i218uiw9w','wjijs')
```

```
INSERT INTO t1(i,c,v) VALUES(-19982921,'s','abc')
```

4.11.3.5.4.1 Set null parameter value (1)

In this example, when exporting the data file in plain text format, the null values in the data are set to the default N.

```
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select * from t1"    -m'0' -e'|$|'     -f'data1.txt'
you machine is Little endian!


   Connecting to test...
   Connected to test.


--- unload [text file] mode ---
--- field="|$|" ---
            0 rows exported at 2013-08-30 13:15:44
            6 rows exported at 2013-08-30 13:15:44
output file t11.dat11 closed
  export:                6 rows.
  export:                4 columns.
  export time:        0.00 sec


   Disconnecting from test...
   Disconnected from test.


$ cat t11.dat11
12|$|34|$|ajhsa                          |$|7shuusa8us
89|$|\N|$|\N|$|\N
0|$|90|$|sniuda                        |$|djkjkdsd
109|$|0|$|                               |$|
850266|$|9055655988595|$|iu92uijw9i218uiw9w    |$|wjijs
-19982921|$|\N|$|s                              |$|abc
```

4.11.3.5.4.2 Set null parameter value (2)

In this example, when exporting the data file in Oracle text format, set the null value in the data to ' N'.

```
$ ./ db2to8a -D'test' -u'db2inst1' -p'db2inst1' -q"select * from t1"    -m'3' -e'|$|'      -n'\\N'
-f'data1.txt'
you machine is Little endian!


   Connecting to test...
   Connected to test.


--- unload [text file] mode ---
--- field="|$|" ---
            0 rows exported at 2013-08-30 13:15:32
            6 rows exported at 2013-08-30 13:15:32
output file t11.dat10 closed
  export:                6 rows.
  export:                4 columns.
  export time:       0.00 sec


   Disconnecting from test...
   Disconnected from test.


$ cat data1.txt
12|$|34|$|ajhsa                        |$|7shuusa8us
89|$|\N|$|\N|$|\N
0|$|90|$|sniuda                       |$|djkjkdsd
109|$|0|$|                             |$|
850266|$|9055655988595|$|iu92uijw9i218uiw9w    |$|wjijs
-19982921|$|\N|$|s                             |$|abc
```

4.11.4 GBaseMigrationToolkit tool usage

GBase Migration Toolkit migration tool is a tool provided by GBase that can realize data migration of heterogeneous databases. At present, data migration in the source database (currently supported source databases include ACCESS, Oracle, SQL Server2005, DM, DB2, MySQL, ShenTong, GBase8sV8.3, GBase8t, GBase8s, PostgreSQL, and Teradata) can be migrated to the target database (currently supported target databases include GBase8a, GBase8t, and GBase8sV8.7).

The migration tool has a simple and easy to operate graphical interface. It can create corresponding tasks according to the requirements of data migration, and set the migration tasks accordingly to achieve multithreading and concurrent data migration.

The migration tool is a C/S structured software that is easy to install and can be used by simply obtaining and decompressing the installation package. (Note: Please do not place it in the Chinese directory, otherwise it may cause some functions to not function properly.)

4.11.4.1 Installation File Description

The GBase Migration Toolkit installation package has versions that match different operating systems and can be obtained as needed. After obtaining the installation package file, extract it and use it:

- Installation package under Linux:

GBaseMigrationToolkit_ 8.5.20.0_ build6_ Linux64.tar.gz

tar -xvf GBaseMigrationToolkit_ 8.5.20.0_ build6_ Linux64.tar.gz

cd GBaseMigrationToolkit_ 8.5.20.0_ build6_ Linux64/jre/bin

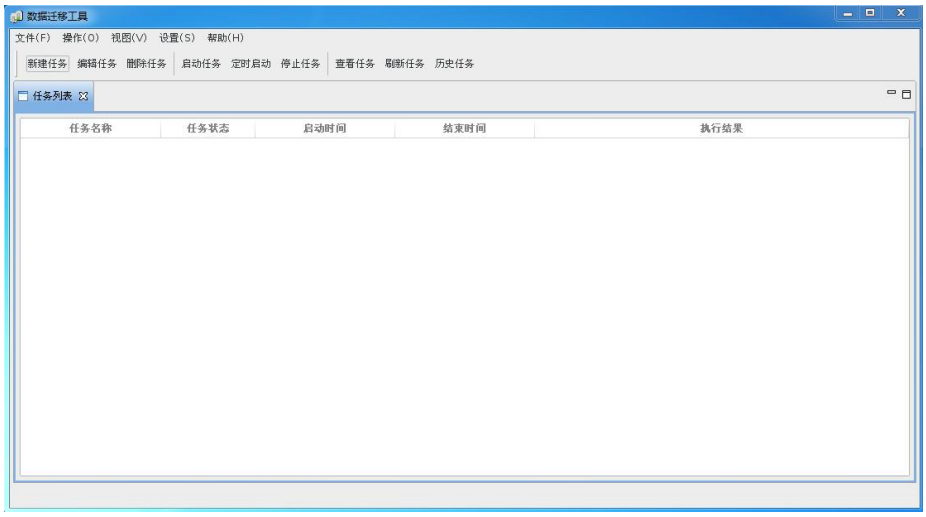
chmod +x *

cd GBaseMigrationToolkit_ 8.5.20.0_ build6_ Linux64/migration

chmod +x Migration

Execute the following command to use:

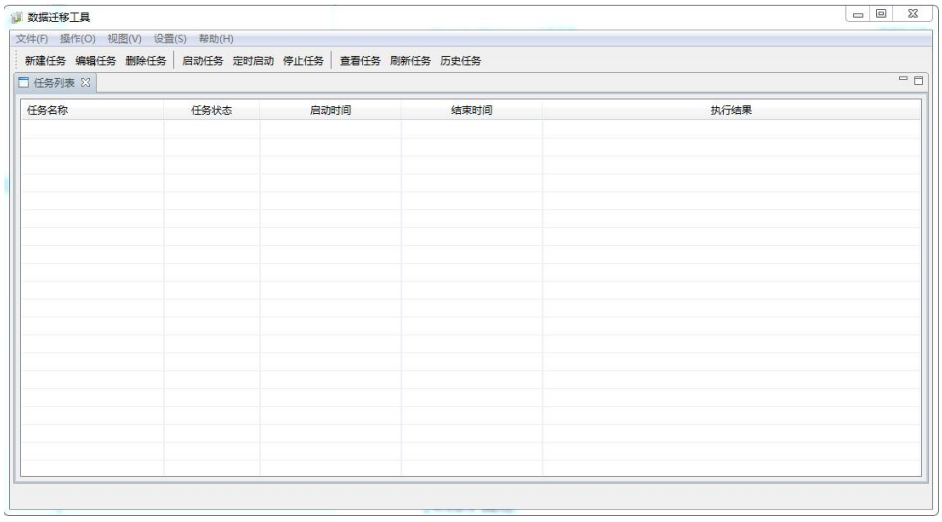
./Migration



- Installation package under Windows:

GBaseMigrationToolkit_ 8.5.20.0_ build6_ winx86_ 64.zip

After decompression, go to the directory migration and execute Migration.exe.



4.11.4.2 instructions

function

The GBase Migration Toolkit supports data migration of heterogeneous databases as follows:

Create migration task: Set the source database, target database, and objects to be migrated. When creating tasks, it is allowed to specify the owner when the source

database is Oracle, and the type of migration table (default random distribution table, replication table, hash distribution table) when the target database is GBase 8a cluster.

Edit Migration Task: Modify and edit the created task

Start Task: Run the specified migration task

Scheduled task: Set a scheduled run for the specified migration task

Stop Task: Abort a running task

Delete Task: Delete the specified task

View Task: View relevant information about the task, including its source library, target library, migration objects, as well as the progress and execution results of the migration

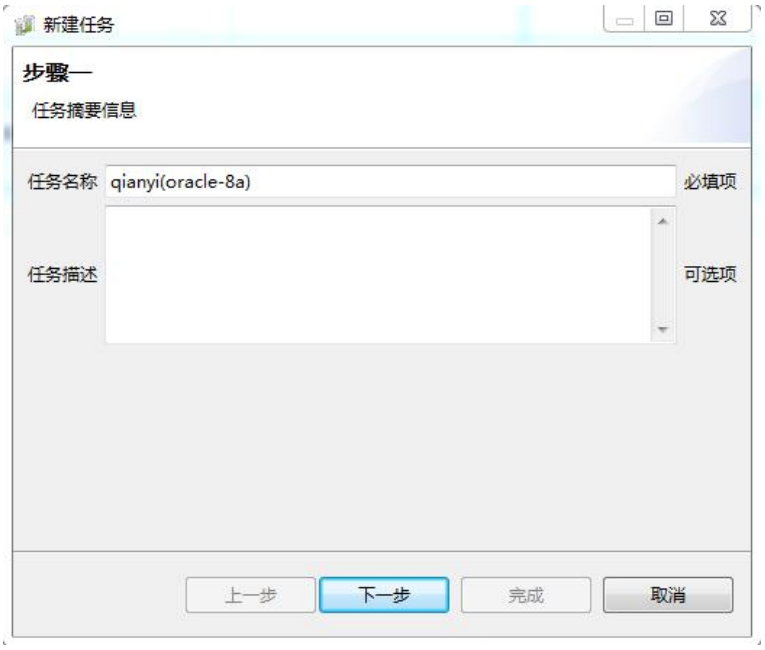
Historical Tasks: View information on executed historical tasks

Settings: Modify the configuration file of the GBase Migration Toolkit tool to customize the number of read and write threads for migration tables and data, as well as the number of data submissions.

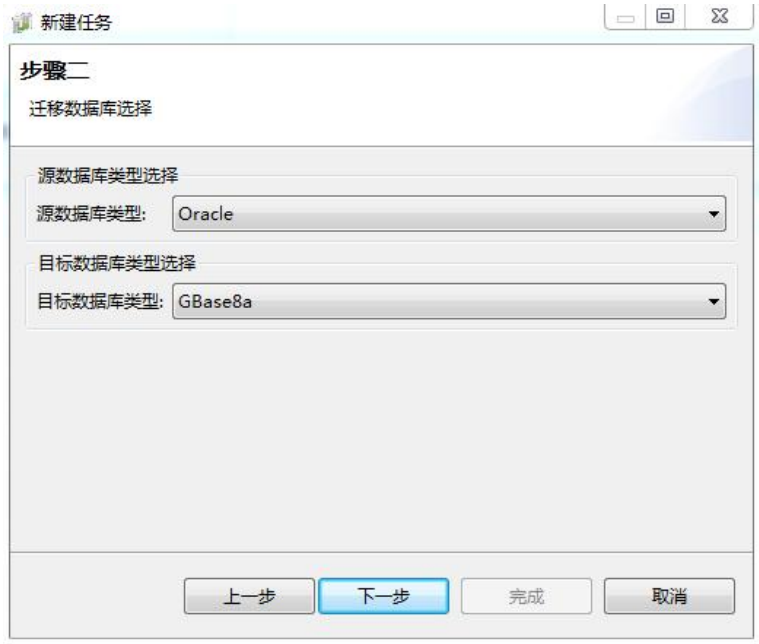
Example

The example of GBase Migration Toolkit migrating Oracle tables to GBase 8a is as follows:

- Step 1: Create a new task

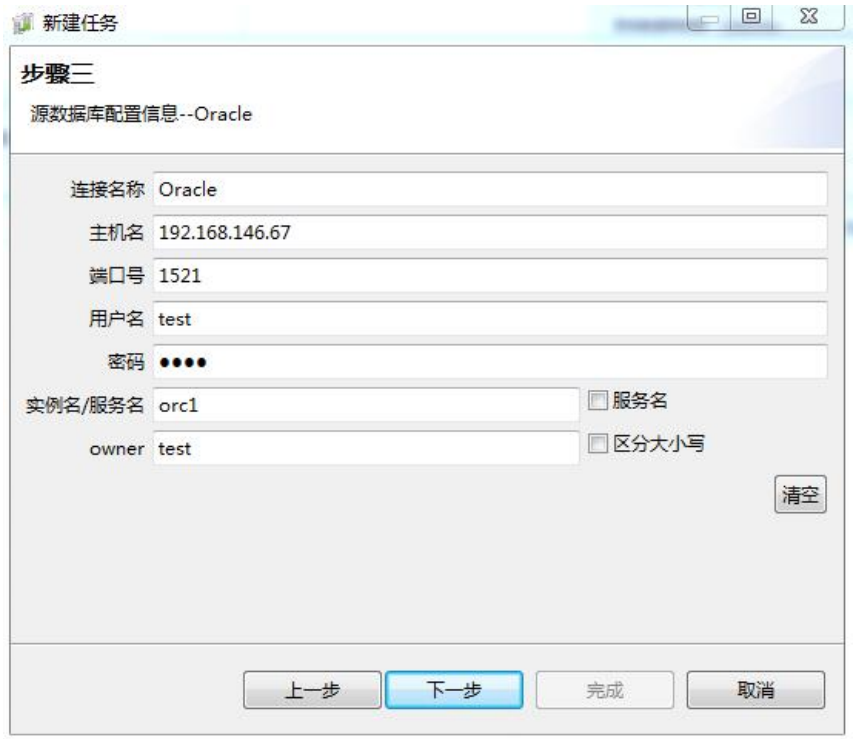


Select source and target libraries



Fill in the connection information for Oracle:

It is necessary to confirm in advance that the Oracle Server firewall has opened the relevant ports or closed the firewall.



Fill in the connection information for GBase 8a:

Select migration object:

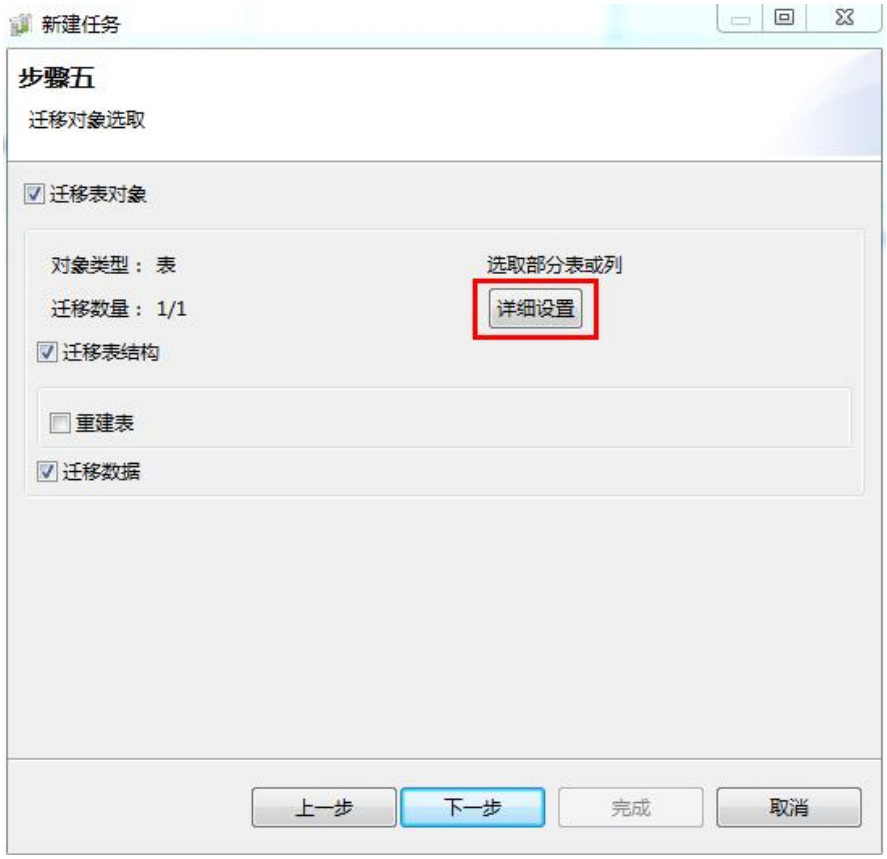
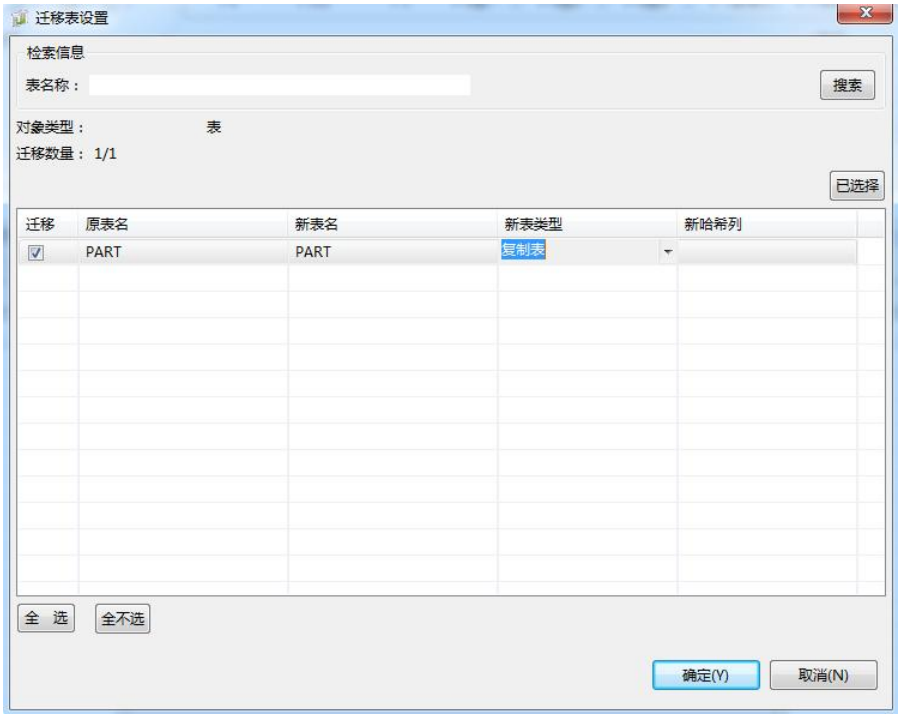
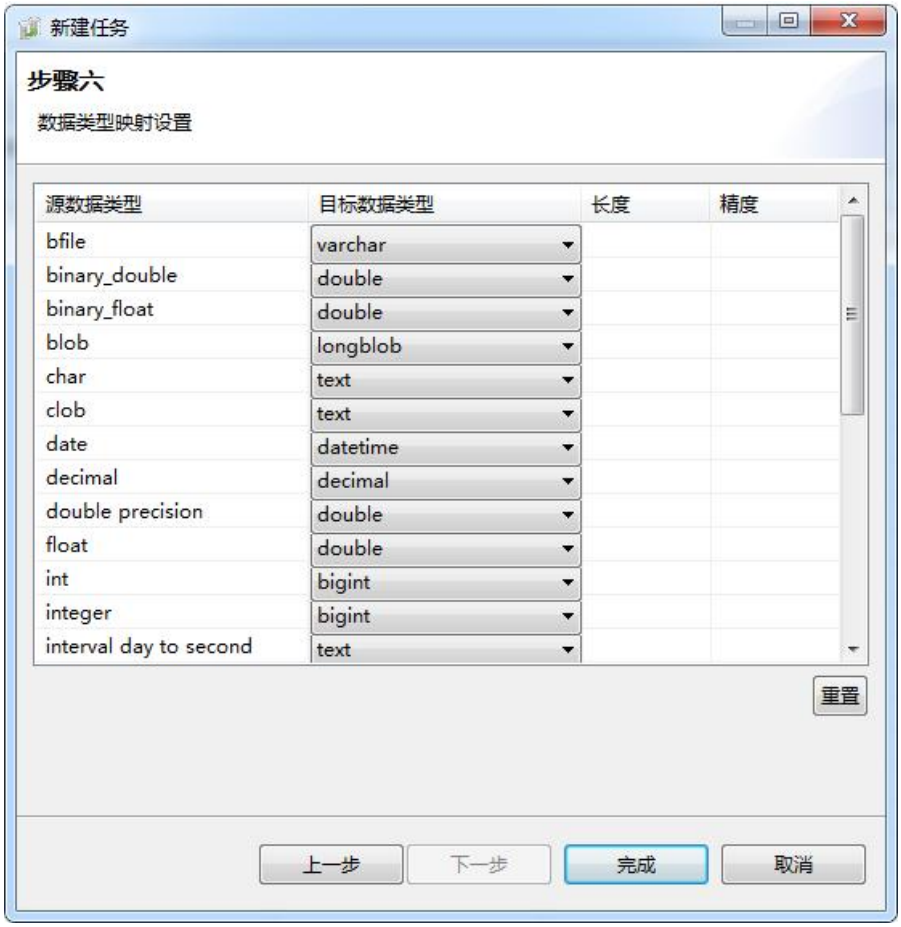


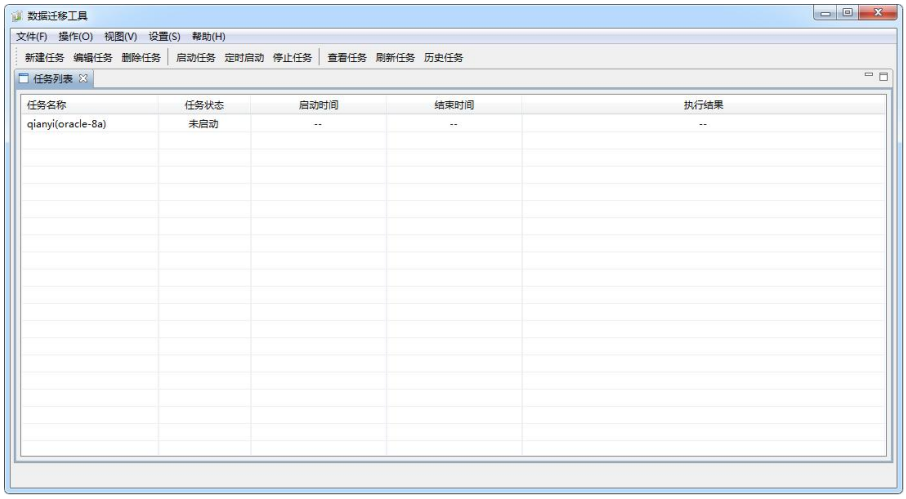
Table type settings for Oracle table migration to GBase 8a:



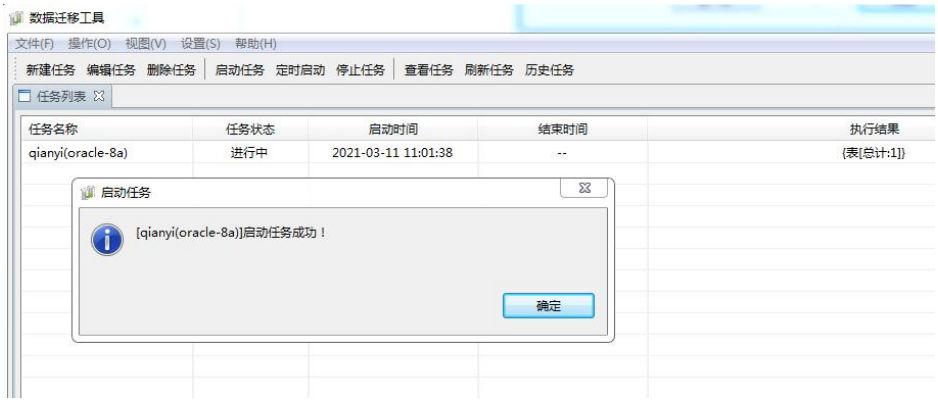
The data type mapping can be adjusted according to the actual situation:



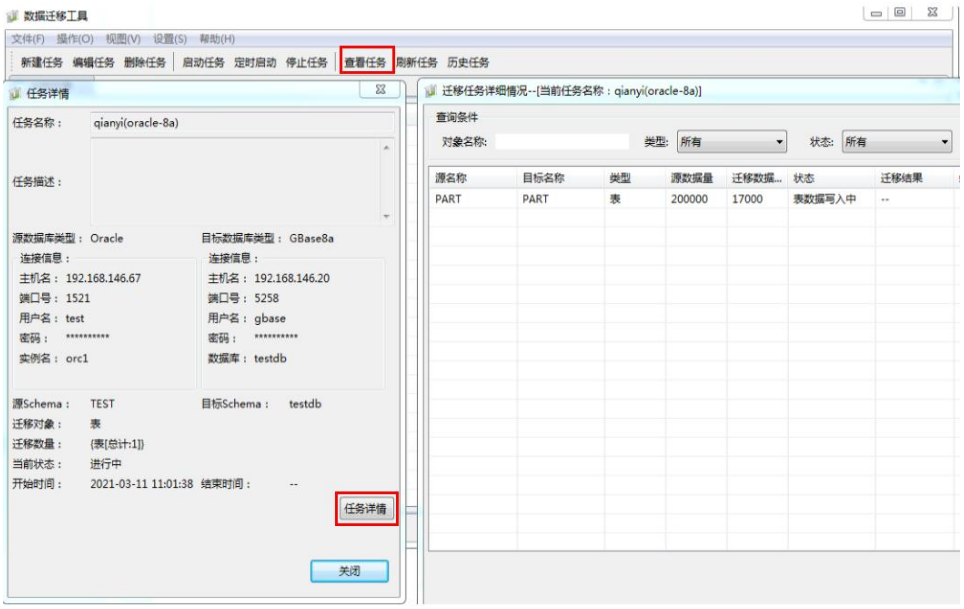
Complete the task creation, and the main interface displays the new task.



- Step 2: Start the task



You can view the progress of tasks by viewing them



4.12Inter cluster synchronization tool

4.12.1 summary

The high availability of GBase 8a MPP Cluster within the cluster is already very complete, and in industries such as finance and security that require higher data security, it requires high availability at the entire instance level. GBase 8a MPP Cluster ensures the high availability requirements of the entire cluster and provides a load balancing mechanism for the entire instance level by establishing another cluster level mirror. GBase 8a MPP Cluster achieves cluster level mirroring through synchronization of underlying binary files.

4.12.2 term

Table -4125 Glossary of Intercluster Synchronization Tools

term	introduce
image	It refers to two clusters with identical structures (number of shards, number of nodes, and distribution).
Distribution	It refers to the consistency of hash values for the same shard of two clusters; Simply understood as the same data, importing on two clusters will fall on the same shard.
A set of available shards	A table has a good shard status in each shard on the cluster (such as n1, n2, n3, n4) (you can use show datacopymap db.tb to view the table shard and its status).
Main cluster	A synchronized source cluster can be understood as a cluster that is in use and needs to be backed up.
Backup cluster	The target cluster for synchronization can be understood as a cluster that serves as a mirror backup.
Main sharding	Main sharding refers to the sharding where the distribution table of a cluster is distributed across various nodes, such as n1->node1, n2->node2, n3->node3, n4->node4. When the cluster runs tasks, the main sharding is preferred.
Backup partition	The backup sharding of the main shard is used to backup the data of the main shard, stored on a different node from the main shard.

4.12.3 Tool installation

- The inter cluster synchronization tool needs to support both incremental and full synchronization methods. Cross cluster synchronization only negative

Synchronize the main sharding and backup sharding settings, relying on the automatic recovery mechanism of the backup cluster for synchronization;

- Install package gcluster before use_ rsynctool-9.5.2.28-redhat7.3-x86_ Simply unzip 64. tar. bz2.

explain

The synchronization tool is divided into v95 and v8 versions. The v95 version synchronization tool only supports cluster synchronization for version 95x, while the v8 version synchronization tool supports cluster synchronization for versions 85 and 86. For example, GBase8a_ MPP_ Cluster-NoLicense-9.5.2.28-redhat7.3-x86_ 64.tar.bz2 gcluster_ rsynctool-9.5.2.28-redhat7.3-x86_ 64.tar.bz2

4.12.4 Explanation of interface and parameter usage scenarios

Command format

```
Usage: gcluster_ rsynctool.py [option]
```

4.12.4.1 Parameter Description

- -h, --help

Meaning: Display tool help information;

Parameter type: optional parameter;

Value range: None;

Precautions and usage restrictions: Specify this parameter to directly display help information before the tool exits.

- -v, --version

Meaning: Display tool version information;

Parameter type: optional parameter;

Value range: None;

Precautions and usage restrictions: Specify this parameter to directly display help information before the tool exits.

- --master_ mpp_ ip=MASTER_ MPP_ IP

Meaning: Used to specify the IP address of any node in the coordinator of the main cluster;

Parameter Type: Required parameter;

Notes and usage restrictions: Only IP addresses in IPV4 format are supported.

- --master_ mpp_ gc_ port=MASTER_ MPP_ GC_ PORT

Meaning: Used to specify the coordinator port information of the main cluster;

Parameter type: optional parameter;

Value range: [default: 5258, min: 1, max: 65536];

Notes and usage restrictions: If the port information is not the default value, please

specify this parameter.

- --master_ mpp_ gn_ port=MASTER_ MPP_ GN_ PORT

Meaning: Used to specify the gnode port information of the main cluster;

Parameter type: optional parameter;

Value range: [default: 5050, min: 1, max: 65536];

Notes and usage restrictions: If the port information is not the default value, please specify this parameter.

- --slave_ mpp_ ip=SLAVE_ MPP_ IP

Meaning: Used to specify the IP address of any node in the coordinator of the backup cluster;

Parameter Type: Required parameter;

Notes and usage restrictions: Only IP addresses in IPV4 format are supported.

- --slave_ mpp_ gc_ port=SLAVE_ MPP_ GC_ PORT

Meaning: Used to specify the coordinator port information of the backup cluster;

Parameter type: optional parameter;

Value range: [default: 5258, min: 1, max: 65536];

Notes and usage restrictions: If the port information is not the default value, please specify this parameter.

- --slave_ mpp_ gn_ port=SLAVE_ MPP_ GN_ PORT

Meaning: Used to specify the gnode port information of the backup cluster;

Parameter type: optional parameter;

Value range: [default: 5050, min: 1, max: 65536];

Notes and usage restrictions: If the port information is not the default value, please specify this parameter.

- --database_ user=DATABASE_ USER

Meaning: Used to specify the database users who connect to the primary and secondary clusters;

Parameter type: optional parameter;

Precautions and usage restrictions: Access to 'table' is required_ list_ The permissions for the table defined in 'file', which

Users need to exist in both the active and standby clusters.

- --master_ mpp_ gc_ pw=MASTER_ MPP_ GC_ PW

Meaning: Used to specify the database user password of the main cluster;

- --slave_ mpp_ gc_ pw=SLAVE_ MPP_ GC_ PW

Meaning: Used to specify the database user password for the backup cluster;

Parameter type: optional parameter;

- --table_ list_ file=TABLE_ LIST_ FILE

Meaning: The configuration file used for the synchronized table.

Parameter Type: Required parameter;

Notes and usage restrictions: The file name, which contains a list of tables that need to be synchronized, can only have

A table. The file content format is DBName.TBName, which does not support the vc.db.tb format. Using line breaks for segmentation, supports both Windows and Linux line breaks, but must be consistent, meaning that only one type of line break can appear in the file.

- --table_ parallel_ degree=TABLE_ PARALLEL_ DEGREE

Meaning: Used to specify the number of tables for each parallel synchronization between clusters;

Parameter type: optional parameter;

Value range: [default: 1, min: 1, max: 128];

Precautions and usage restrictions: This parameter needs to be adjusted based on the load of the active and standby clusters, as well as the parallelism of the business

Section; When the job parallelism is low and the load on the active and standby clusters is not high, the configuration parameter can be appropriately increased.

- --lock_ table_ timeout=LOCK_ TABLE_ TIMEOUT

Meaning: Used to specify the timeout period for the inter cluster synchronization tool to lock the primary and backup cluster tables

Parameter type: optional parameter;

Value range: [unit: second, default: 600, min: 1, max: 3600];

Notes and usage restrictions: Since locks are used for write operations on mutually exclusive tables, this parameter needs to be based on the table's

Extend the maximum write operation time appropriately.

- --retry_ times=RETRY_ TIMES

Meaning: Used to specify the number of retries after synchronization failure between the underlying shards and shards;

Parameter type: optional parameter;

Value range [unit: times, default: 1, min: 1, max: 10]

Precautions and usage restrictions: This parameter mainly depends on the network status. In case of poor network status (flickering, network blocking), sharding to sharding synchronization failure may occur. A retry is required to ensure successful synchronization, and appropriate adjustments need to be made based on the network status.

- --retry_ interval=RETRY_ INTERVAL

Meaning: Used to specify the time interval between each retry after the synchronization failure of the underlying sharding pair shards;

Parameter type: optional parameter;

Value range: [unit: second, default: 10, min: 1, max: 1800];

Notes and usage restrictions: This parameter mainly depends on the network status. In cases of poor network status (flash, network blockage), there may be synchronization failure between shards. After the failure, it is necessary to wait for a period of time to wait for the network to recover, and then try again. This parameter is only an empirical parameter and cannot guarantee that the network will recover completely and the synchronization will be successful after waiting for a period of time.

- --sync_ mode=SYNC_ MODE

Meaning: Used to specify the mode of data synchronization [rough filtering, incremental synchronization, full synchronization];

Parameter type: optional parameter;

Value range [unit: none, default: 1, min: 1, max: 3].

explain

The meaning of parameter values and precautions and usage restrictions for usage scenarios:

Rough filtration

Meaning: Check the change identification of the table. If the change identification of the table in the active and standby clusters is the same, skip the table directly and do not synchronize; The change identification of the table is different, and then check the change identification at each column level for incremental synchronization;

Applicable scenario: This parameter value is applicable to the synchronization of incremental data after the initialization of the backup cluster is completed. In this case, it is recommended to use this parameter.

Incremental synchronization

Meaning: Do not check the change identification of the table, directly check the change identification of each column level, and perform incremental synchronization;

Applicable scenario: This parameter value is applicable to the synchronization of incremental data after the initialization of the backup cluster is completed. This parameter is reserved due to version iteration history and is purely for version compatibility.

Full synchronization

Meaning: Do not perform any level of change identification checks, directly overwrite the data of the backup cluster with the data of the host cluster;

Applicable scenario: This parameter value is applicable when it is necessary to manually cover all data of the backup cluster, which generally occurs in the following situations:

The table of the backup cluster has been manually modified, and the data is no longer trusted;

The table data of the backup cluster is damaged and needs to be rebuilt and repaired;

Initialize the backup cluster.

- --error_ table_ list_ file=ERROR_ TABLE_ LIST_ FILE

Meaning: Used to specify the file stored in the failed synchronization table;

Parameter type: optional parameter;

Value range: [default: current directory/${table_list_filename}_error_table_list_% Y_% m_% d -% H:% M:% S. log];

Notes and usage restrictions: The file content format is DBName.TBName, separated by line breaks; Optional parameter, the default value of the parameter is the current directory, and the default name is ${table_list_filename}_ error_ table_ list_% Y_% m_% D -% H:% M:% S.log, with content format of db.tb, one per line, stored in multiple lines; This file is overwritten every time the tool is called.

- --log_ name=LOG_ NAME

Meaning: Used to specify the storage file for tool logs;

Parameter type: optional parameter;

Value range: [default: current directory/gcluster_rsynctool_yyyy_mm_dd. log];

Precautions and usage restrictions: The specified directory location must have write permission for the tool execution user.

- --log_ level=LOG_ LEVEL

Meaning: Used to specify the tool log level;

Parameter type: optional parameter;

Value range [default: 3, min: 0, max: 5].

explain

Precautions and usage restrictions:

nolog level;

critical level;

error level;

warning level ;

info level;

debug level;

- --rsync_ mode=RSYNC_ MODE

Meaning: Used to specify the tool scheduling mode;

Parameter type: optional parameter;

Value range [default: 2, min: 0, max: 2].

explain

Precautions and usage restrictions:

0: Main sharding synchronization, backup sharding setting status

Meaning: Only synchronize one set of shards of the backup cluster table, and set the status of other backup shards to be restored through the automatic recovery mechanism within the cluster;

Applicable scenario: This parameter value is not recommended and is only reserved for version compatibility;

1: Simultaneous synchronization between master and backup

Meaning: Simultaneously synchronizing the primary and backup shards of the backup cluster, maximizing the performance of single table synchronization;

Applicable scenario: This parameter value is not recommended. There is a situation where both the master and backup of a shard fail to synchronize and the table is unavailable. This parameter was originally designed for scenarios with low data security.

2: First active backup synchronization method

Meaning: First synchronize the main shard of the backup cluster table, and then synchronize the backup shard after the main shard synchronization is successful, ensuring that there is a set of available shards in the backup cluster table after synchronization failure, which is used to rollback synchronization operations;

Applicable scenario: It is recommended to use this parameter to ensure the data security of the backup cluster table.

- --double_ check

Meaning: Used to specify whether to enable read back verification;

Parameter type: optional parameter;

No parameter, value range [default: false];

Precautions and usage restrictions: After data is written to the disk, read back to check the backup cluster table data; This parameter will reduce the performance of synchronization and increase the disk IO consumption of the backup cluster; It can be used in the early stages of deployment to verify the correctness of synchronization.

- --slave_ create_ table_ if_ not_ Exists optional parameter

Create table if not slave mpp. [default:false]

Precautions for parameter usage:

After enabling this parameter, it will use the database specified by the inter cluster synchronization tool when the slave needs to create a table_ When a user creates a table, if the table on the master is not created by the user, it can cause exceptions on the slave. The exceptions include but are not limited to: unsuccessful table creation, permissions issues after the table is created, unexpected UID of the table, disk space limitations in resource management errors, etc. When enabling this parameter, users need to strictly follow that the synchronized tables are all specified databases_ Table of users.

- --sync_ vc_ name

VC names that need to be synchronized.

Note:

Synchronizing one VC content at a time does not support synchronizing information from multiple VCs simultaneously;

Synchronous VC requires consistent VCname and table distribution, and does not synchronize mirror table content;

When synchronizing a table, only the data and metadata information of the table are synchronized, and no other content is synchronized.

4.12.4.2 Multiple instance IPMAPPING

The inter cluster synchronization tool supports mapping synchronization during multi instance deployment.

- IPmapping function description:

The IP mapping function is mainly used in scenarios where the IP address used for synchronizing data between the primary and backup clusters is different from the IP address used for installation in the primary and backup clusters. Generally, the installation of IP in a cluster uses a private network, and the synchronization of data between clusters uses a dedicated public IP address for data synchronization. The public and private networks are not accessible.

The inter cluster synchronization tool is implemented through GC of the backup cluster_ sync_ GC of client and main cluster_ sync_ Server completes data synchronization

point-to-point, cluster node GBase service, GC_ sync_ Client services, GC_ sync_ The server service uses cluster installation IP, so using IP mapping for synchronization requires the data synchronization IP of each node to correspond to the installation IP. In a multi instance scenario, a physical machine has multiple instances, and each instance has its own installation IP. However, multiple instances on a physical machine share an external synchronization IP. IP mapping requires that the installation IP of each instance be mapped to the unique "data synchronization IP+port" of each instance, so that the "public IP+port" can access the corresponding unique instance in the cluster. Namely, IPMapping needs to enable network interoperability as shown in the following figure:

[Main Cluster] Instance 1 Private IP "--" [Main Cluster] Public IP+Instance 1gnode External Port+Instance 1sync External Port "--" [Backup Cluster] Public IP+Instance 1gnode External Port+Instance 1sync External Port "--" [Backup Cluster] Instance 1 Private IP

- Operation method:

  1) Modify the synctool configuration file $GBASE for all gnodes in the active and standby clusters_ BASE/config/syncool. conf, add external binding IP and syncServer external port, restart syncServer service

  Add the following parameters to synctool. conf:

  BIND_ ADDRESS_ ADDICTIONAL=Public External IP

  SERVER_ PORT_ ADDICTIONAL=syncServer External Port

  Note: BIND_ ADDRESS_ ADDICTIONAL and SERVER_ PORT_ ADDICTIONAL cannot be associated with an existing BIND_ The ADDRESS and 5288 ports are identical at the same time.

  2) Modify the gbase configuration file $GBASE for all gnodes in the active and standby clusters_ BASE/config/gbase_ 8a_ Gbase.cnf, add external binding IP and gbase external port, restart gbase service

  gbase_ 8a_ Add the following parameters to gbase.cnf:

  bind_ address_ Additional=Public network external IP

  port_ Additional=gbase External Port

  Note: bind_ address_ Additional and port_ Additional cannot be associated with an existing bind_ The address and port 5050 are identical at the same time

3) Inter cluster synchronization tool gcluster_ Gcluster after rsynctool decompression_ Configure the master in the rsynctool directory_ Mapping and slave_ Mapping configuration file

```
$ cat master_ mapping
{
    "OWNER":"MASTER",
    "IPMAPPING":[
        {
"From": "Primary cluster instance 1 internal IP",
"TO": "Primary cluster instance 1 external IP",
            GNPORT ":" Primary cluster instance 1 gbase external port ",
            SYNCPORT ":" Primary cluster instance 1 synctool external port "
        },
        {
"From": "Main cluster instance 2gnode internal IP",
"TO": "Primary cluster instance 2 external IP",
            GNPORT ":" Main cluster instance 2 gbase external port ",
            SYNCPORT ":" Main cluster instance 2 synctool external port "
        }
      ... ...
    ]
}
$ cat slave_ mapping
{
    "OWNER":"SLAVE",
    "IPMAPPING":[
        {
"From": "Internal IP address of backup cluster instance 1",
```

```
"TO": "External IP address of standby cluster instance 1",
        GNPORT ":" Backup cluster instance 1 gbase external port ",
        SYNCPORT ":" Alternate cluster instance 1 synctool external port "
    },
    {
"From": "Internal IP address of standby cluster instance 2gnode",
"TO": "External IP address of standby cluster instance 2",
        GNPORT ":" Backup cluster instance 2 gbase external port ",
        SYNCPORT ":" Alternate cluster instance 2 synctool external port "
    }
  ... ...
 ]
}
```

Note: The external IP in the mapping must be consistent with the external binding IP of the previous gbase and syncserver, and the port must also be consistent. The port of GNPORT and gbase must also be consistent_ Additional consistency, SYNCPORT and SERVER_ PORT_ ADDICTIONAL is consistent.

4) Start the rsynctool and use it normally.

- Example:

There are two primary and secondary clusters, each with two physical machines and two instances on each physical machine. The following is the configuration of four instances of two physical machines in the main cluster. The backup cluster structure is consistent with the main cluster, and the configuration method is also consistent with the main cluster. The following example will not be described again.

Main cluster:

Physical machine 1 external IP: 100.100.100.150

1）gnode1：

Internal IP: 192.168.8.147

Synctool. conf Add:

BIND_ ADDRESS_ ADDITIONAL=100.100.100.150

SERVER_ PORT_ ADDITIONAL=5289

gbase_ 8a_ Gbase.cnf Add:

bind_ address_ additional=100.100.100.150

port_ additional=5051

2）gnode2：

Internal IP: 192.168.8.148

Synctool. conf Add:

BIND_ ADDRESS_ ADDITIONAL=100.100.100.150

SERVER_ PORT_ ADDITIONAL=5290

gbase_ 8a_ Gbase.cnf Add:

bind_ address_ additional=100.100.100.150

port_ additional=5052

Physical machine 2 external IP: 100.100.100.155

1）gnode1：

Internal IP: 192.168.8.150

Synctool. conf Add:

BIND_ ADDRESS_ ADDITIONAL=100.100.100.155

SERVER_ PORT_ ADDITIONAL=5291

gbase_ 8a_ Gbase.cnf Add:

bind_ address_ additional=100.100.100.155

port_ additional=5053

2）gnode2：

Internal IP: 192.168.8.151

Synctool. conf Add:

BIND_ ADDRESS_ ADDITIONAL=100.100.100.155

SERVER_ PORT_ ADDITIONAL=5292

gbase_ 8a_ Gbase.cnf Add:

bind_ address_ additional=100.100.100.155

port_ additional=5054

$ cat master_ mapping

```
{
    "OWNER":"MASTER",
    "IPMAPPING":[
        {
            "FROM":"192.168.8.147",
            "TO":"100.100.100.150",
          "GNPORT":"5051",
          "SYNCPORT":"5289"
        },
        {
            "FROM":"192.168.8.148",
            "TO":"100.100.100.150",
          "GNPORT":"5052",
          "SYNCPORT":"5290"
        },
       {
            "FROM":"192.168.8.150",
            "TO":"100.100.100.155",
```

```
            "GNPORT":"5053",
            "SYNCPORT":"5291"
        },
    {
            "FROM":"192.168.8.151",
            "TO":"100.100.100.155",
        "GNPORT":"5054",
        "SYNCPORT":"5292"
        }
    ]
}
```

4.12.5 Example

Scenario Description

Number of nodes: 2 nodes in the main cluster, single sharding, one backup; Backup two nodes in the cluster, single sharding, and one backup. rsync_ Mode=0, do not specify double_ check。

Main and backup:

```
drop database if exists test;
create database test;
use test;
create table t(a int)distributed by ('a');
```

Main:

```
insert into t values(1),(2),(3),(4),(5),(6),(7),(8),(9),(0);
insert into t values(1),(2),(3),(4),(5),(6),(7),(8),(9),(0);
insert into t values(1),(2),(3),(4),(5),(6),(7),(8),(9),(0);
```

```
insert into t values(1),(2),(3),(4),(5),(6),(7),(8),(9),(0);

insert into t values(1),(2),(3),(4),(5),(6),(7),(8),(9),(0);

insert into t select * from t;

insert into t select * from t;

insert into t select * from t;

insert into t select * from t;

insert into t select * from t;
```

Tools:

```
t.list:

test.t
```

Execute Command

```
./gcluster_ rsynctool.py --master_ mpp_ ip=192.168.3.180 --slave_ mpp_ ip=192.168.3.182 --table_
list_ file=t.list --log_ level=5    --rsync_ mode=0
**********Gcluster Sync Tool Start**********
Table [                    test:                    t] Sync Start
Table [                    test:                    t] Sync End         cost : <0 s,597 ms>
**********Gcluster Sync Tool End With Success**********

Query backup cluster
gbase> select count(*) from t;
+----------+
| count(*) |
+----------+
|         1600 |
+----------+
1 row in set (Elapsed: 00:00:00.01)
```

4.13 Transparent gateway tool between clusters

4.13.1 Preconditions for use

- Currently, only the Linux operating system is supported;
- Support heterogeneous data sources such as Oracle, MySQL, and Teradata;
- The user who starts the transparent gateway has read and write permissions to GBase8a MPP Cluster through the transparent gateway;
- When gcluster_ dblink_ direct_ data_ When the exchange parameter is set to 1, it is necessary to ensure that the cluster versions of the source database and the target database are consistent.
- Dblink currently supports homologous character set access for homogeneous data sources: gbk<->gb18030, utf8<->utf8mb4. The default character set for 953 is utf8mb4862.33. The default character set is utf8, and dblink currently supports connected access from V953 to V862 versions and from V953 to V952 versions.
- Dblink does not support an ipv4 network as the source and an ipv6 network as the target, which may result in an error. The source end is an ipv6 network, and the target end is an ipv4 network that can support dblink.
- Dblink does not support the target table being a version that supports autoincrement columns and has autoincrement columns, while the source table is a version that does not support autoincrement columns. For example, 953 does not support tables with self increasing columns pulling data from 862 through dblink.

4.13.2 Installing a transparent gateway

The installation steps for a transparent gateway are as follows

Step 1

After obtaining the tar package of the GBase 8a MPP Cluster transparent gateway, copy the tar package to the target installation path.

Step 2

Use the following command to decompress. After successful decompression, a directory with the same name as the tar package will be generated in the current path. This directory is the installation directory for the transparent gateway.

```
Tar - xvf [compressed package name]. tar
```

Step 3

Use the following command to grant the current user corresponding permissions to files and subdirectories in the transparent gateway installation directory.

```
Chmod - R+x [installation directory name]
```

4.13.3 Configurating Transparent Gateways

The gateway service configuration file is as follows:

(The storage path is in the conf folder under the directory generated after extracting the installation package, which is [Gateway Installation Directory]/conf/):

- Gateway parameter configuration file: conf.properties
- Gateway data source configuration file: dataSource/sample/dblink_ name.properties
- Gateway target database configuration file: gcluster/gbase8a_ gcluster.properties

Table -4126 Parameter Description of Conf.properties Configuration File

Configuration Item	Is it mandatory	notes	Default value
gbase.gt.port	yes	Define the binding port for listening services	nine thousand eight hundred and ninety-eight
gbase.gt.encode	yes	Define the coding format of interactive information to avoid garbled codes	utf-8
gbase.gt.pagesize	yes	Sql Page Size per Page	three thousand
gbase.gt.load.data.type	yes	The method of loading data, batch method is 1, and insert values method is 0	0
gbase.gt.table.use.decimal	yes	Is the decimal type used when the length of an integer value exceeds 19 bits	one
gbase.gt.gc.paging.query	yes	When the source database is	0

Configuration Item	Is it mandatory	notes	Default value
		gcluster, do you want to use pagination queries? 0 is non pagination, and 1 is pagination	
gbase.gt.wait.timeout	yes	Connection timeout waiting time (seconds)	seven thousand and two hundred

Table -4127 dblink_ Parameter Description of name.properties Configuration File

Configuration Item	Is it mandatory	notes	Default value
dataSource_ IP	yes	IP address of the data source GBase cluster	—
dataSource_ port	yes	Port of the source database	—
dataSource_ dbname	yes	The name of the source database	—
dataSource_ dbtype	yes	The type of source database, currently gcluster must be filled in	—
dataSource_ user	yes	The username of the source database	—
dataSource_ pwd	yes	Password for the source database	—
dataSource_ charset	no	The character set of the source database Used to set the character set encoding used by the target end to send SQL to the source end	By default, use the system character set of the local DB (character_set_system)

Table -4128 gbase8a_ Gcluster.properties configuration file parameter description

Configuration Item	Is it mandatory	notes	Default value
gcluster_ IP	yes	IP address of the target GBase cluster	—
gcluster_ port	yes	Port of the target database	—
gcluster_ encode	yes	Encoding of the target database The configuration is based on the expected character set encoding that the target end expects to receive from the source end	—
gcluster_ user	yes	The username of the target database	—

Configuration Item	Is it mandatory	notes	Default value
gcluster_ pwd	yes	Password for the target database	—

explain

gbase8a_ The target configuration item for gcluster.properties is 8.5.1.2_ Build 4.8 and above.

Example of gateway parameter configuration file

The gateway configuration file is fixed to conf.properties. The example is as follows:

(Path: [Installation directory]/conf/conf. properties)

```
gbase.gt.port=9898
gbase.gt.encode=utf-8
gbase.gt.pagesize=100000

gbase.gt.load.data.type=0
gbase.gt.gc.paging.query=0
gbase.gt.wait.timeout=7200
```

Example of gateway data source configuration file

Multiple data sources can be configured, each corresponding to a configuration file, identified by the data source name, such as the data source dblink_ The configuration file name for name is dblink_ Name.properties, an example of the configuration file content is as follows:

(Path: [Installation directory]/conf/datasource/)

```
[ds1]
dataSource_ IP=192.168.8.102
dataSource_ port=5258
dataSource_ dbname=dtest
dataSource_ dbtype=gcluster
dataSource_ user=gbase
dataSource_ pwd=******
```

be careful

The current version only supports gcluster and oracle data source types.

Example of gateway target database configuration file

Multiple target libraries can be configured, with each target corresponding to a configuration file, identified by the target name:

(Path: [Installation directory]/conf/gcluster/gbase8a_gcluster. properties)

```
[gc1]
gcluster_ IP=2001:da8:e000:0:1:1:5
gcluster_ port=5258
gcluster_ encode=utf-8
gcluster_ user=gbase
gcluster_ pwd=******
```

be careful

- gcluster_ The IP needs to be configured with a complete address, and the 0 in the middle of the address cannot be omitted
- Target configuration file support 8.5.1.2_ Build4.8 and above
- To install the source and target clusters with a character set of gbk, the parameter dataSource needs to be set_ Charset and gcluster_ Encodes are all gbk.

4.13.4 Start transparent gateway

Enter the gbaseGateway directory and execute the following command to start the transparent gateway:

```
$ sh gt.sh
```

4.13.5 Uninstall Transparent Gateway

- The transparent gateway of GBase 8a MPP Cluster cluster is pure portable application, which is not intrusive to the operating system, so you do not need to use the uninstall program to uninstall it, just delete the entire deployment directory of the transparent gateway directly;
- Enter the directory above the transparent gateway installation directory, execute rm - r installation directory name, and then delete the installation directory.

4.13.6 Deployment Example

4.13.6.1 Deployment Environment

- Operating system: redhat 6;
- Database: GBase8a MPP Cluster database.

4.13.6.2 Configurating Transparent Gateways

Step 1

Install a transparent gateway and unzip the tar package

```
Tar - xvf [installation package]
```

Step 2

Modify the corresponding permissions of the transparent gateway:

```
Chmod - R+x [installation directory]
```

Step 3

Configure a transparent gateway and modify the system parameter configuration file: [Installation directory]/conf/conf.properties, with the following modifications:

```
$ cat conf.properties
gbase.gt.port=9898
gbase.gt.encode=utf-8
gbase.gt.pagesize=10000
gbase.gt.load.data.type=0
gbase.gt.gc.paging.query=1
```

```
gbase.gt.wait.timeout=7200
```

Step 4

Modify the gateway data source configuration file: [installation directory]/conf/datasource/gc_ Link1. properties, the content is as follows:

```
$cd [installation directory]/conf/datasource/
$ cp sample/gbase_ link1.properties    gc_ link1.properties
$ cat dataSource/gc_ link1.properties
[gbase8a cluster]
dataSource_ IP=[Data Source IP]
dataSource_ port=5258
dataSource_ Dbname=[Database name of the data source]
dataSource_ dbtype=gcluster
dataSource_ user=gbase
dataSource_ pwd=******
```

Step 5

Modify the gateway target database configuration file: [installation directory]/conf/gcluster/gbase8a_ Gcluster.properties, the content is as follows:

```
$ cat gcluster/gbase8a_ gcluster.properties
[gc1]
gcluster_ IP=192.168.146.20
gcluster_ port=5258
gcluster_ encode=utf-8
gcluster_ user=gbase
gcluster_ pwd=******
```

Step 6

Start transparent gateway

```
$cd [installation directory]

$ sh gt.sh
```

Step 7

Transparent gateway shutdown method. If the transparent gateway configuration fails, the corresponding process number needs to be manually killed, and then reset

New Launch

```
ps -ef|grep GBaseGateway
```

```
Kill [process number]
```

4.13.6.3 Configure target database GBase8a MPP Cluster A

Configure the configuration file parameters for the target database

(Path: $GCLUSTER_BASE/config/gbase_8a_gcluster. cnf for each management node in the target database)

gbase_ 8a_ The parameters that must be configured in gcluster.cnf:

- gbase_ dblink_ gateway_ IP=[Transparent gateway IP, such as 192.18.16.11]
- gbase_ dblink_ gateway_ Port=[Service port of the transparent gateway, such as 9898]

gbase_ 8a_ Optional parameters in gcluster.cnf:

- gcluster_ dblink_ direct_ data_ exchange

explain

Default value is 1

- A value of 1 indicates that data is directly sent from the computing node of the data source cluster to the target cluster;
- A value of 0 indicates that data is sent from the data source cluster to the gateway, which then forwards it to the target cluster.

> **be careful**
>
> Adjust this parameter to 0 only when the nodes of the data source cluster and target cluster cannot be directly connected on the network.

4.13.6.4 Using db link

Create db link

Grammar format

```
CREATE DATABASE LINK dblink_ name connect to '' identified by '' using 'gc_ link ';
```

explain

- Among them, dblink_ Name is the name of a custom db link, which will be used for db link queries in subsequent queries. Connect to " identified by " is a fixed grammar;
- The name of the data source follows using. For example, the data source configuration file name is gc_ Link. properties, then fill in using 'gc' here_ link'。

Delete db link

Grammar format

```
DROP DATABASE LINK dblink_ name;
```

explain

The IF EXISTS syntax is currently not supported.

Using db link

Grammar format

```
SELECT * From table name @ dblink_ name
```

4.14DBLink tool

4.14.1 Manage DBLink

Manage DBLink, including configuring gbGateway information, creating, deleting, and querying DBLink.

Configure gbGateway information

- Configure the gbGateway information to be used by GBase 8a MPP Cluster (local).
- The GBase 8a cluster (local) uses the name of the external data source and the corresponding relationship between the gbGateway and the gbGateway.

explain

The content includes the name of the external data source, the IP address of the gbGateway, and the port number of the gbGateway.

Create DBLink

Create a DBLink pointing to gbGateway on GBase 8a MPP Cluster (local). Admin can

point to SQL statements that create DBLinks through any GBase 8a MPP Cluster (local) application.

Grammar format

```
CREATE DATABASE LINK dblink_ name CONNECT TO username IDENTIFIED BY password USING 'TG_ config_ name';
```

explain

All users under the current database service can use or delete this DB Link;

- dblink_ Name: The name of the DB Link to be created;
- Username: The username in the database service that the DB Link wants to connect to;
- Password: The username and password in the database service that the DB Link wants to connect to;
- TG_ config_ Name: The name of the transparent gateway configuration file.

Example

```
create database link dblink_ pub
connect to sysdba identified by sys
using 'tg_ config1';
```

Delete DBLink

Delete DBLink pointing to gbGateway on GBase 8a MPP Cluster (local). Admin can point to the SQL statement to delete DBLink through any GBase 8a MPP Cluster (local) application.

Grammar format

```
DROP DATABASE LINK dblink_ name;
```

Example

```
drop database link dblink_ pub;
```

Query DBLink

View all dblinks that have been created, and the information for each dblink includes:

- dblink_ Name: The name of the DB Link to be created

- Username: The username in the database service that the DB Link is connecting to
- Password: Display as a NULL value
- TG_ config_ Name: The name of the transparent gateway configuration file.

4.14.2 Using DBLink

GBase 8a cluster (local) queries can use dblinks that have already been created.

Grammar format

```
table@dblinkname
```

Example

```
select * from    table@ dblink_ pub;
```

 **explain**

Create a view. If the definition statement of the view includes a dblink query, it is necessary to create a view in gcluster

Configure the following parameters in the gclusterd configuration file of COORDINATOR:

```
gbase_ dblink_ standby_ gateway_ ip

gbase_ dblink_ standby_ gateway_ port
```

4.14.3 Terminate DBLink query

There are two ways for users to terminate DBLink queries. One is that users can directly use the Ctrl+C combination key on the query execution console (the session where the query is executed); Secondly, on other consoles, send kill sessions_ The method of the id command.

The effect of terminating a query is discussed in two situations.

- For select * from t1@gc_link t1, t2@gc_link t2 where t1.a = t2.a; Similar to this pure homologous DBLink table query, the query results are directly returned to the user. The command to terminate the query can provide a quick response. When the user initiates an operation to terminate the query, the execution of the remote (data source) will also be interrupted to quickly respond to the user's operation;

- For other types of DBLink queries (such as insert into t select * from t1@gc_link t1, t2@gc_link t2 where t1.a = t2.a;) 。 When the purpose of remote (data source) queries is to pull data to the local cluster, the remote (data source) cannot quickly respond to user termination query operations. If the current step is a remote (data source) query, the user's query termination operation needs to wait for the remote (data source) execution to end before the query can end.

explain

When the user's DBLink query is terminated, the following error message will be printed on the corresponding console:

```
gbase> select * from lineitem@tpch_link l, orders@tpch_link o ;

ERROR 1317 (70100): Query execution was interrupted
```

4.14.4 Syntax constraints for db link queries

The syntax constraints for db link queries are as follows

- The db link table can only appear in top-level queries or subqueries of dblinks from the same source. Appear in local table

  When in a sub query of, it must be placed in a relationship sub query.

  - For example, the following statement will report a syntax error, because when a db link appears in a subquery of a local table, it must be surrounded by a relation subquery.

    ```
    select * from t1 where exists (select 1 from t2@gc_link as t2 where t2.id = t1.id);
    ```

  - This statement can be modified to the following form to ensure compliance with grammar rules:

    ```
    select * from t1 where exists (select 1 from (select 1 from t2@gc_link ) as t2 where t2.id = t1.id);
    ```

- Homogeneous db link tables can be JOIN directly. The dblink table is prohibited from being associated with local tables, relationship subqueries,

  Non homologous dblink tables generate direct JOIN relationships.

- For example: t1@gc_link JOIN t2@gc_link It is allowed; but t1@gc_link JOIN t2 is not allowed because the db link table prohibits direct JOIN with the local table. The SQL statement can be rewritten as follows:

```
… (select * from t1@gc_link ) t, t2 …
```

- Local tables and non homologous dblink tables are prohibited from appearing in sub queries of the db link table.

- For example: select * from t1@gc_link where exists (select 1 from t2); It is not allowed because the db link table t1@gc_link Local table t2 appears in the subquery of. It can be rewritten as follows:

```
select * from (select * from t1@gc_link ) t where exists (select 1 from t2);
```

- Related sub queries in group by or order by are prohibited from appearing in the db link table.

be careful

- If it is only for simple use, such as inserting into local_ Table select * from homologous dblink

Table, the constraints in this section can be ignored. When there is a mix of db link tables and local tables in the query (example

If the database link table JOIN is a local table, or when mixed with non homologous database link tables, it is necessary to

Observe the following syntax constraints, or the query will report a syntax error;

- Homologous db links refer to db links with the same name but different names as non homologous

db-link。 for example t1@gc_link and t2@gc_link Two db-links considered to be homologous

Table. t1@gc_link and x1@gc_link2 Is considered a non homologous db link table, even if

gc_ Link and GC_ Link2 was created using the same data source due to its db link

Different names are still considered non homologous.

4.14.5 Instructions for using private dblink

Usage Scenario

DB created by 1_ Link, hoping to be used by specific users, other users cannot call it casually, that is, it supports private db_ link.

2. In order to improve db_ It is recommended to send the performance of the link locally to the remote SQL through dblink, without parsing locally, and directly to the remote for execution. This supports the passthrough mode: passthrough

3. Support private dblinks in direct mode.

Example

Create a private dblink for cluster B, for u_ Dblink user private:

create user u_ dblink; u_ The dblink permissions are as follows:

grant all on gctmpdb.* to u_ dblink;

grant all on dbname.* to u_ dblink;

grant select on gbase.* to u_ dblink;

gccli -u u_ dblink

create private database link db_ linkBtoA connect to '' identified by '' using 'dsBtoA';

Only u in cluster B_ Dblink can use db_ linkBtoA

A sends a command to B to extract data from A to B. The following SQL is issued from cluster A:

A pushes data to B's table:

insert into tb_B@db_linkAtoB select * from dbname.tb_ A

A sends a command to B to pull data to B's table:

passthrough link db_ linkAtoB using 'insert into dbname.tb_ B select * from tb_A@db_linkBtoA ;

Grammar:

1. Support private dblink syntax

create private database link ...

drop private database link ...

2. Do not modify default behavior

create database link ... , Default is public link

Create public database link, specify as public link

3. Using dblink in direct mode

passthrough link publiclink using 'insert into t1 select * from t2@privatelink '

Cluster 1 maintains public link, while cluster 2 maintains private link. Cluster 1 initiates a direct mode command, and then executes the pull table t2 data from cluster 1 and inserts it into the t1 table of cluster 2 on cluster 2. The private dblink of cluster 2 is only available to the created user, and cannot be used by other users.

be careful:

When configuring publiclink, specify that the user accessing cluster 2 in the configuration file should be the same as the privatelink user created on cluster 2.

Need to add parameters to the configuration files of all management nodes in the cluster

gbase_ 8a_ gcluster.cnf

gbase_ dblink_ gateway_ IP=Gateway IP

gbase_ dblink_ gateway_ port=9898

4.14.6 Heterogeneous data sources

The DBLink gateway supports heterogeneous data sources such as Oracle, MySQL, and Teradata, mainly responsible for metadata data type mapping and data extraction. GBase 8a MPP Cluster serves as the entry point for dblink queries, responsible for SQL parsing of dblink queries, interacting with gateways (obtaining metadata, requesting data extraction, etc.), generating query plans, executing plans, scheduling execution, and other tasks.

Attention should be paid to the use of heterogeneous data sources:

Oracle default object names are all uppercase, while MySQL object names are case sensitive by default. When using these heterogeneous data sources, it is necessary to set gcluster appropriately_ dblink_ orcl_ case_ The sensitive parameter value prevents

DBLINK from being unable to find the specified object in heterogeneous data sources due to object name case issues.

gcluster_ dblink_ orcl_ case_ When the sensitive value is 0, DBLINK will convert the table name to uppercase. When the value is 1, DBLINK does not convert case, which is consistent with SQL writing. The default value of this parameter is 0.

4.14.6.1 Gateway Data Source Configuration

Example

Example of dblink gateway oracle/mysql/teradata data data source:

```
Cp [Gateway installation directory]/conf/dataSource/sample/oracle_ Link1. properties [Gateway installation directory]/conf/dataSource/oracle_ link1.properties
cat oracle_ link1.properties
[ds1]
dataSource_ IP=192.168.6.124-- The IP address of the host where the Oracle/mysql/teradata service is located
dataSource_ Port=1521-- The port on which the Oracle/mysql/teradata service listens
dataSource_ Dbname=orcl -- Oracle/mysql/teradata library name
dataSource_ Dbtype=oracle/mysql/teradata -- The data source type is oracle/mysql/teradata
dataSource_ User=myora -- oracle/mysql/teradata username
dataSource_ Pwd=myora -- password for oracle/mysql/teradata user
```

explain

The data source needs to be compatible with Oracle 11g and 10g (the gateway needs to be compatible).

The Teradata data source needs to add the following parameters:

dataSource_ url=jdbc: teradata://IP/TMODE=ANSI ,CHARSET=UTF8,database=***

Please refer to section 4.13.3 for the configuration of other configuration files for the gateway.

4.14.6.2 Dblink query syntax constraints

Add syntax constraints for heterogeneous data sources:

- Except for union (also including mini, union all, intersect) queries, heterogeneous data source dblink tables are only allowed to appear in from sub queries. Assuming that olink is a heterogeneous data source dblink object, such as the following SQL:

```
select * from t1@olink ;          -- Non compliant
Error message: DBLink table from heterogeneous data source must be long to the relationship subquery
select * from (select * from t1@olink );    -- accord with
select * from t1@olink    tt1 join t2@olink Tt2 on tt1. a=tt2. a -- Not compliant
Error message: DBLink table from heterogeneous data source must be long to the relationship subquery
select * from (select * from t1@olink    tt1 join t2@olink    tt2 on tt1.a=tt2.a) ttt ; -- accord with
select * from t1@olink    tt1 union select * from t1@olink    tt2;    -- Compliant with
select * from (select * from t1@olink    tt1 union select * from t1@olink    tt2) ttt; -- accord with
select tt1.a from t1 tt1 join (select a from t1@o_link ) tt2 on tt1.a=tt2.a; -- accord with
select * from (select a from t1@o_link ) tt1
join (select a from t1@o_link ) tt2 on tt1.a=tt2.a; -- accord with
select tt1.a from t1@o_link    tt1 join (select a from t1@o_link ) tt2 on tt1.a=tt2.a; -- Non compliant
Error message: DBLink table join with (normal table | | from sub query) is forbidden
select * from t1 where a in (select a from t1@o_link );.....-- Non compliant
Error message: The position of DBLink table is in a subquery of normal table is forbidden
select * from t1 where a in (select a from (select a from t1@o_link )tt); -- accord with
select * from t1@o_link    tt1 union all select * from t1@o_link    tt2; -- accord with
select * from t1@o_link    tt1 minus select * from t1@o_link    tt2; -- accord with
select * from t1 tt1 union select * from t1@o_link    tt2; -- accord with
select * from t1@o_link    tt2 intersect select * from t1 tt1; -- accord with
```

4.14.6.3 Oracle/mysql/teradata syntax compatibility

There is no support for dialects and proprietary functions of Oracle/mysql/teradata.

explain

For heterogeneous data source Oracle:

- Parameters_ t_ gcluster_ dblink_ clear_ syntax_ Constraints=2: does not support using group in dblink queries_ Concat function.

- Parameters_ t_ gcluster_ dblink_ clear_ syntax_ Constraints=1: Supports using group in dblink queries_ Concat function.

4.14.6.4 Metadata compatibility

The large object data types (such as blob, clob, etc.), binary, and long of Oracle/mysql/teradata are not supported in this issue. Supports Oracle basic data types (character, numeric, and date).

explain

- The decimal (p, s) of gbase, where p refers to the maximum precision supported to 65 digits and s refers to the decimal, with a maximum value of 30; If the number (p, s) type of Oracle exceeds 30, the dblink gateway will map to the double data type of GBase. As the double type is an imprecise data type, there will be a loss of accuracy;
- Due to the fact that the date type of Oracle can store time, minute, and second information, for heterogeneous data sources, the date type of Oracle will be mapped to the date time type of GCluster. Using dblink to query oracle's date type data will bring time, minute, and second information. You can use to_ The date function formats the output to remove time, minute, and second information.

4.14.6.5 character set

Capable of supporting Oracle/mysql/teradata with UTF8 and GBK/GB2312 character sets, and supporting Chinese data sources.

4.14.6.6 Scenarios supported by SQL functionality

The scenarios supported by dblink's heterogeneous data source SQL function are consistent with those of homogeneous data sources:

- Support select statements to query dblink tables;
- Support for create... select, The select section is queried using dblink;
- Support insert select .., The select section is queried using dblink;
- Support multi table association delete, and use dblink queries for the source part;
- Support the use of dblink queries in stored procedures;

- Support the prepare statement for preprocessing dblink queries.

Example

```
set @sql_ str='select id2 from x1@gc_dblink    where id2=1';

prepare stmt from @sql_ str;

execute stmt;

deallocate prepare stmt;
```

4.14.6.7 Scenarios not supported by SQL functionality

The following SQL functions are consistent with homogeneous data sources and are currently not supported:

- DDL operations on dblink tables, such as drop table, alter table, etc., are not supported;
- Creating views using dblink queries is not supported;
- It is not supported to use dblink tables or queries as source associations for updating local tables;
- The merge statement using the source part using dblink query is not supported;
- The use of dblink queries in functions is not supported.

Example

Example 1

DDL operations on dblink tables, such as drop table and alter table, are not supported:

```
drop table x1@gc_dblink ;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your GBase server version for the right syntax to use near '@gc_ dblink' at line 1
```

Example 2

The update/delete/insert/merge operation on the dblink table is not supported:

```
update x1@gc_dblink    set id2 =1;
ERROR 1105 (HY000): DBLink-Table does not support update operation.
```

```
delete from x1@gc_dblink ;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your GBase server version for the right syntax to use near '@gc_ dblink' at line 1
```

Example 3

Creating views using dblink queries is not supported:

```
create view v1 as select * from t1@testlink ;
ERROR 1235 (42000): This version of GBase doesn't yet support 'use dblink table in a FUNCTION/TRIGGER/VIEW.'
```

Example 4

Updating local tables using dblink tables or queries as source associations for updates is not supported

```
update t1, t1@o_link    tt1 set t1.b=tt1.b where t1.a=tt1.a;
update t1 join t1@_link    tt1 on t1.a=tt1.a set t1.b=tt1.b;
update t1 join (select a,b from t1@_link ) tt1 on t1.a=tt1.a set t1.b=tt1.b;
ERROR 1105 (HY000): DBLink-Table does not support update operation.
```

Example 5

The merge statement using dblink query in the source section is not supported:

```
merge into x1 tt1 using (select * from x1@olink ) tt on (tt1.id2=tt.id2)
when matched then update set tt1.id3=tt.id3
ERROR 1105 (HY000): DBLink-Table does not support update operation.
```

Example 6

The use of dblink queries in functions is not supported:

```
delimiter //
create function dfunc(id int) returns int
begin
declare fid int default 1;
set fid = (select id2 from x1@olink    where id2=id limit 1);
return fid;
end //
This version of GBase doesn't yet support 'use dblink table in a FUNCTION/TRIGGER/VIEW.
```

4.14.7 DBLINK data push

- The target table that supports the insert... select statement is the dblink remote table, which supports pushing local data to the remote through the insert... select statement;
- When pushing local data to a remote Oracle table, it should be noted that the Oracle character type is in bytes, while the GBase character type is in characters. Therefore, when the GBase source table and the Oracle target table contain the same type of field char (255), pushing the data of that field to Oracle may result in an insert time out of bounds situation;
- At the same time, it supports the passthrough mode, which means that GCluster does not parse the specified SQL and requests the gateway to directly forward the specified SQL to the remote database for execution. Through the passthrough mode, operations such as insert... values, delete, update, etc. can be performed on the remote table.

4.14.7.1 Data Push SQL Statement

- Grammar and insert into... select The statement is the same, and the target table is supported as a dblink remote table, such as:

  insert into t1@testlink select a, b from t1;

- Support specifying target columns, such as:

  insert into t1@testlink (a, b) select a, b from t1;

- Prerequisite: The user of the data source configuration in the gateway corresponding to the dblink object must have access to the target table

  Insert permission.

- Only auto commit mode is supported, and Distributed transaction is not supported. If the current session of gcluster is executed

  If the status is non automatic submission mode, an error will be reported:

  ```
  Can not join the distributed transaction in session
  ```

- It is necessary to ensure the atomicity of a push SQL statement. The data can be pushed to the remote end or all

Part failed.

 **be careful**

- The empty string data of GBase 8a MPP Cluster is pushed to ORACLE through dblink, and ORACLE is stored as NULL, meaning ORACLE does not distinguish between empty strings and NULL, and can be queried using is null;
- GBase 8a MPP Cluster distinguishes between empty strings and NULL.

4.14.7.2 Direct Mode SQL Statement

Request the gateway to directly forward SQL statements to the remote database corresponding to the dblink object for execution.

Grammar format

```
passthrough link DBLINK_ NAME using ‘SQL_ STATEMENT’;
```

explain

- DBLINK_ NAME is the name of the dbink object;
- SQL_ Statement is an SQL statement, which is the SQL statement executed by the remote database;
- Supported SQL statements, except for the following supported SQL types, other types of SQL will report errors:

```
insert into … values…
insert into …select ….
delete
update
truncate
merge
create
drop
```

Example

```
passthrough link testlink using ‘create table t1(a int, b int)’;
passthrough link testlink using ‘insert into t1 values(1,2)’;
passthrough link testlink using ‘update t1 set a=11 where a=1’;
passthrough link testlink using ‘delete from t1 where a=11’;
passthrough link testlink using ‘truncate table t1’;
passthourgh link testlink using ‘drop table t1’;
Passthrough link testlink using 'select * from t2' - Error: SQL command is not supported: 'select * from
```

```
t2'
```

explain

Constraints and limitations:

- The user configured for the data source in the gateway corresponding to the dblink object must have corresponding SQL execution permissions;
- Only automatic submission mode is supported. If the session status of the gclusterd currently executing the passthrough command is in non auto commit mode, an error will be reported:

```
Can not join the distributed transaction in session
```

- SQL not supported_ There is a comment before the statement.

4.14.7.3 Support redundant deployment of gateway HA

- Support redundant deployment of gateway high availability;
- Support the deployment of two sets of gateways with the same configuration, one primary and one backup. If the primary gateway cannot be connected, it will automatically connect to the backup gateway;
- The gcluster dblink function supports high availability and redundant deployment of dblink gateways, with the following new parameters added:
  - gbase_ dblink_ standby_ gateway_ IP: The IP address of the host where the backup dblink gateway service is located;
  - gbase_ dblink_ standby_ gateway_ Port: The port on which the standby dblink gateway service listens.
- Two sets of dblink gateway services with the same configuration can be deployed, one active and one standby, both of which are started online. When gcluster fails to connect to the main gateway, it will attempt to connect to the backup gateway;
- Supported data sources:
  - Support homogeneous data source gcluster;
  - Heterogeneous data sources only support Oracle.

4.14.7.4 Data type compatibility

- Large object data types (blob, clob, etc.) and binary data types are currently not supported;
- For heterogeneous data sources, basic data types (such as character, number, and date) are supported, and data that meets the representation range and format of the target data type can be correctly pushed to the remote end.

explain

- Oracle big object data types and binary types include Blob, clob, nclob, bfile, Raw, long Raw
- Oracle supports data types such as Char, nchar, varchar, varchar2, nvarchar2, Number, Integer, Float, Binary_ float, Binary_ double, Date, timestamp

4.14.7.5 Character set support

- Capable of supporting data sources from UTF8 and GBK/GB2312 character sets, as well as Chinese data sources;

4.14.8 DBLINK parameter configuration

4.14.8.1 _ t_ gcluster_ having_ without_ group_ by

_ t_ gcluster_ having_ without_ group_ The by parameter is used to set whether to support having statements without group by.

- The default value is 0, indicating that it is not supported;
- When set to 1, it supports having statements without group by.

Note: For dblink queries, this parameter needs to be set to the global level or added to the configuration file to take effect.

explain:

Global level variables_ t_ gcluster_ having_ without_ group_ By=1, the scenario of homogeneous data sources is as follows:

1. There is only one step in dblink, where the executor directly connects to the remote SQL and uses asynchronous APIs to pass the local session variable to the remote end. Therefore, this variable will also be set, and having without group by can succeed. For example:

   select sum(i) from tt1@gc_dblink having sum(i) ;

2. The dblink part of the SQL is sent to the remote cluster for execution. This part of the SQL does not have group by but has having. Due to the remote cluster being 8a and not having this parameter enabled, an error will be reported as a failure. as

   select sum(i) from tt1@gc_dblink having sum(i) union select sum(i) from s1@gc_dblink having sum(i);

   ERROR 1105 (HY000): (GBA-02SC-1001) The query includes syntax that is not supported by the gcluster.

4.15Node fault detection and recovery tools

Node Fault Detection and Recovery Tool: GC_ node_ detect.py

Function Description

The tool reads SQL execution information from the specified table (sys_sqls_elapsepernode) every other period of time (default 30 minutes), and this information is recorded in the tool's log (gc_node_detect. log). The tool uses the database table GC_ node_ detect_ The error information keywords provided in errmsg are used to count the number of specific SQL errors, and corresponding strategies are selected and executed based on the number of specific SQL errors and the policies in the configuration file.

- Configuration file gc_ node_ Detect. conf description:

The above detection time period, detection interval, threshold of error SQL number, and error node processing strategy can all be configured according to actual needs in the configuration file of the tool (gc_node_detect. conf).

If the threshold for the number of SQL statements with specific errors is set to 5 in the configuration file, and the policy is to send email reminders and offline the error nodes, when the number of SQL statements with specific errors reaches 5, the policy will be triggered to send emails to the DBA and offline the error reporting nodes.

- Node offline description:

Node offline is to set the status of the node to failure. Before performing the offline

operation, it will first determine whether the node is the unique shard of a table. If so, the node cannot perform the offline operation.

- Table GC_ node_ detect_ Errmsg description:

Users can store incorrect keywords that require targeted capture into GC based on actual situations_ node_ detect_ Errmsg table, gc_ node_ The detect tool compares the error SQL information read from the system table with the GC_ node_ detect_ Compare the error information in the errmsg table and calculate the GC_ node_ detect_ Process the number of incorrect SQL statements specified in the errmsg table.

- Log description:

$GCLUSTER_ BASE/log/gcluster/gc_ node_ detect.log

Zhizhong recorded GC_ node_ The detect tool regularly reads SQL execution status information from system tables, including information on the SQL with specified errors, percentage information, execution strategy information, etc.

- System Table sys_ sqls_ Elapsepernode Description

The execution information of the historical SELECT is saved in the gclusterdb library, sys_ Sqls and sys_ sqls_ In the elapsepernode table, the execution information of the currently executing select is saved in the INFORMATION_ SCHEMA.SELECT_ In the STATISTIC table

gcluster_ dql_ statistic_ After the threshold parameter is enabled, sys_ Sqls and sys_ sqls_ The elapsepernode table will record TASK information that exceeds the time specified by this parameter, and the system table sys_ sqls_ The elapsepernode has an error message field errmsg for select execution. The information recorded in this table can be used to identify the error node and further determine whether there is a related hardware fault on the node and whether a certain strategy is triggered.

sys_ Sqls and sys_ sqls_ Elapsepernode table in gcluster_ dql_ statistic_ After the threshold parameter is enabled, it will be created and will not be cleaned up. After the parameter is closed, it will not be deleted. You need to manually set the event to regularly clean up the data of these two tables.

Tool runtime environment:

Tool Path: $GCLUSTER_ HOME/bin/

1) Tools require a Python 2. x environment

2) The tool needs to be deployed on the coordination management node (gcware node)

3) The tool relies on sys under the gclusterdb library_ Sqls and sys_ sqls_ Elapsepernode table, gcluster needs to be configured according to actual situation_ dql_ statistic_

Threshold parameter, TASK exceeding the specified time will be recorded in sys_ Sqls and sys_ sqls_ In the elapsepernode table.

- Tool Launch:

User initiated, running in the background

cd $GCLUSTER_ HOME/bin/

./gc_ node_ detect.py &

- Tool stop:

kill -9 `pidof gc_ node_ detect.py`

- Tool instructions:

1) The tool is currently in an inactive state by default, and it is currently not highly available. If the tool stops abnormally, manual detection and processing are required, which can be done through pidof gc_ node_ Detect.py to check if the tool is running.
2) If the tool detects sys while in use_ Sqls and sys_ sqls_ The elapsepernode table does not exist and will prompt an error message, ignoring this detection. The tool will not stop running and will wait for the next detection time.
3) When the tool is in use, if the configuration file does not exist or the configuration information is missing, an error message will be displayed to complete the configuration file, and the tool will stop running.
4) When the tool is in use, connecting to the GBase library is the internal interface of gcware that is called. If the connection fails, an error will be reported and the tool will stop running.
5) If the tool fails to execute the policy during use, error information will be recorded in the log, and the tool will not stop running.

- Tool configuration:

$GCLUSTER_ BASE/config/gc_ node_ detect.conf

By default, scan the SQL execution status within 60 minutes every 30 minutes

The content of the configuration file is as follows. If the parameter values in the configuration file are illegal, the tool will report an error and exit. By default, the configuration file will scan sys every 30 minutes_ Sqls and sys_ sqls_ The elapsepernode table records the data generated within 60 minutes.

[user_basic]

mail_ server= gbase@gbse.com --Mail server

sender_ mail= from@gbase.com --Email sender email

receivers= dba1@gbase.com , dba2@gbase.com --Receive email, multiple configurable, separated by commas

[statistic_strategy]

polling_ time_ Interval=1800-- Polling time interval, in seconds

sql_ time_ Interval=3600-- SQL statistics time period, in seconds

alarm_ Threshold=10-- The absolute threshold for the number of specified error SQL statements that trigger the alarm policy. This value cannot be a percentage, for example, if the absolute number of specified error SQL statements is greater than or equal to 10, the alarm can be triggered

[exception_strategy] - Exception handling module, specifying an exception handling method, with a value of 1 indicating that the method is enabled

is_ Offline=1-- Whether to offline the problem node

is_ Mail=1-- Whether to send an email to notify the DBA of this exception

4.16List of high-risk operations

Table -4129 describes the high-risk operations that should be noted during the operation and maintenance phase of GBase 8a MPP Cluster.

Table -4129 High risk operations

Operation classification	Operation Name	Operational risk	Avoidance measures
install	Installation path mount configuration	It may cause the Gbase 8a MPP Cluster to fail to start.	The installation path mount configuration needs to be written in/etc/fstab instead of/etc/rc.d/rc.local to ensure that the installation path is completed before the Gbase 8a MPP Cluster starts, and to ensure that the mount configuration remains valid after the server restarts.
	Multiple network card network settings	There is an issue with synchronization between nodes.	When setting up multiple network cards, the IP address of each card needs to be configured in different network segments.
	Fingerprint acquisition	It may cause some node services to fail to start.	Ensure that the fingerprint file contains the fingerprint information of all cluster nodes.
Node Replacement	Set the node state to UNAVAILABLE state	This operation is irreversible. After the node state transitions to UNAVAILABLE, the node state can only transition to ONLINE when the node replacement is successful.	Make sure to replace the node before setting the UNAVAILABLE state for the node.

5 Database Management Guide

This chapter provides guidance on the daily management and maintenance of the GBase 8a MPP Cluster database.

5.1 SQL Language Reference

5.1.1 Specification Introduction

5.1.1.1 Identifier syntax rules

The names of objects such as databases, tables, columns, and aliases are all referred to as identifiers. This section describes the syntax rules allowed for identifiers in GBase 8a MPP Cluster.

Table 51 describes the maximum allowed length and characters that can be used for each type identifier.

identifier	Maximum length (characters)	Characters allowed
database	English 48 Chinese 48	A-z, A-Z, 0-9, and underline must start with a letter or underscore, and support Chinese databases.
surface	English 56 Chinese 21	A~z, A~Z, 0-9, underline, Chinese, must start with a letter or underscore, and supports Chinese tables.
view	English 56 Chinese 50	A-z, A-Z, 0-9, and underline must start with a letter or underscore, and support Chinese views.
column	English 64 Chinese 64	A-z, A-Z, 0-9, -, underline, Chinese, must start with a letter or underscore, and supports Chinese columns.
alias	English 256 Chinese 85	A-z, A-Z, 0-9, and underline must start with a letter or underscore, and support Chinese aliases.
stored procedure	English 64 Chinese 64	A-z, A-Z, 0-9, and underline must start with a letter or underscore, and support Chinese stored procedures.
User variables		Composed of a-z, A-Z, 0-9, and an underscore, it must start with a letter or an underscore. User variable names are not case sensitive.

be careful

1、 Except for the limitations specified in the table, identifiers cannot contain ASCII (0) or ASCII (255). Database, table, and column names should not end with spaces.

2、 If the identifier is a restrictive word or contains special characters, when users use it, they must always refer to it with '', such as: SELECT * From 'select'. id>100.

3、 If the identifier length exceeds the maximum length limit, commands for databases, tables, columns, views, and stored procedures will report errors, and aliases will be truncated to 256 characters for display.

4、 In the actual application system, the identifier cannot use the reserved word of GBase 8a MPP Cluster, nor can it contain special characters. For the reserved word supported by the GBase 8a MPP Cluster database, see the reserved word of the GBase 8a MPP Cluster analytical database in section 5.1.2.

5.1.1.2 Identifier qualifier

GBase 8a MPP Cluster allows names to be composed of one or more identifiers. The components of a combination name should be separated by the English period character '.'. The beginning of a combination name is used as a qualifier, which affects the interpretation of subsequent identifiers in the context.

In GBase 8a MPP Cluster, users can refer to a column using any of the following tables:

Table 52 Column Reference Method

Column references	meaning
col_ name	Col_ The name comes from the corresponding field in any table used for the query.
table_ name.col_ name	Col_ Name comes from the table in the current database_ name。
database_ name.table_ name.col_ name	Col_ Name from database_ Table in name_ name。
vc_ name.database_ name.table_ name.col_ name	Col_ Name from virtual cluster VC_ Name's database_ Table in name_ name。
`column_ name`	This field is a keyword or contains special characters.

If a combination identifier needs to be referenced, each part of the identifier shou ld be referenced separately, rather than referring to the combination identifier as a whole. For example: 'gs table'` The 'gs column' is legal, while the 'gs table. gs column' is illegal.

There is no need to explicitly specify a table in the column reference of a statement_ name 、 database_ name.table_ Name or VC_ name.database_ name.table_ The name prefix, unless there is ambiguity in this reference. For example:

1）Assuming tables t1 and t2 both contain a field c, when using a SELECT that uses t1 and t2 to retrieve c, field c is ambiguous because it is not unique in the table used in this statement. Therefore, it is necessary to indicate which table the user needs by writing t1. c or t2. c.

2）If retrieving from table t in database db1 and table t in database db, the user must use db1. t.col_ Name and db2t.col_ Name to specify which library table column to reference.

3）If retrieving from table t of database db1 in vc1 and table t of database db in vc2, the user must use vc1. db1. tcol_ Name and vc2.db2.t.col_ Name to specify which library table column to reference.

5.1.1.3 Annotation syntax

GBase 8a MPP Cluster supports three annotation styles:

'#': single line comment;

"--": A single line comment, with the comment content starting with "--" and ending at the end of the line. Note that the "--" (guide number) annotation requires at least one space after the second guide number;

/* Comment Content */": This type of annotation supports the comment content to be one line or multiple consecutive lines, and also supports the comment content to be in the middle of the line/**/ This closed sequence may not necessarily be represented on the same line, so this syntax allows for multiple line comments.

be careful

- The annotation style of "--" (guide number) requires at least one space after the second guide number. This syntax is slightly different from the standard SQL annotation style.

Example

Example 1: Using a '#' annotation.

```
gbase> SELECT 1+1 FROM t;# This comment continues to the END of line
+-----+
| 1+1 |
+-----+
|   2 |
+-----+
```

```
1 row in set
```

Example 2: Use a "--" annotation.

```
gbase> SELECT 1+1 FROM t;-- This comment continues to the END of line
+-----+
| 1+1 |
+-----+
|   2 |
+-----+
1 row in set
```

Example 3: Use the "/* single line */" annotation.

```
gbase> SELECT 1 /* this is an in-line comment */ + 1 FROM t;
+--------+
| 1    + 1 |
+--------+
|        2 |
+--------+
1 row in set
```

Example 4: Use "/* multiple lines */" comments.

```
gbase> SELECT 1+
    /*
      this is a
      multiple-line comment
    */
      1 FROM t;
+--------------+
| 1       +        1 |
+--------------+
|                   2 |
+--------------+
1 row in set
```

5.1.1.4 User variables

GBase 8a MPP Cluster supports user variables. The lifecycle of user variables is session level and is not visible to other sessions. When a user exits, all user variables for that user will be automatically released.

The writing rule for user variables is: @ var_ name。

Define and assign values to variables through SET syntax:

SET @var_ name = expr [, @var_name = expr] ...

'=' is an assignment operator. The expr value assigned to each variable can be a real number, string, or NULL.

To view the values of user variables through the SELECT syntax:

SELECT @var_ name [, @var_name] ...

Example

Example 1: Using the SET statement to assign values to variables.

```
gbase> SET @t1='abc',@t2=null,@t3=4;
Query OK, 0 rows affected

gbase> SELECT @t1,@t2,@t3;
+------+------+------+
| @t1  | @t2  | @t3  |
+------+------+------+
| abc  | NULL |    4 |
+------+------+------+
1 row in set
```

User variables can be used anywhere the expression allows. If the variable used by the user is not initialized, its value is NULL.

be careful

- Variables cannot be used in the context of constants, for example, in the LIMITED clause of a SELECT.

5.1.2 SQL standards followed

Introduce the development history of SQL standards and the SQL standards supported by GBase 8a MPP Cluster.

A Brief History of SQL Development

The brief history of SQL development is as follows:

1986, ANSI X3.135-1986, ISO/IEC 9075:1986, SQL-86;

1989, ANSI X3.135-1989, ISO/IEC 9075:1989, SQL-89;

1992, ANSI X3.135-1992, ISO/IEC 9075:1992, SQL-92 (SQL2);

1999, ISO/IEC 9075:1999, SQL: 1999 (SQL3);

In 2003, ISO/IEC 9075:2003, SQL: 2003 (SQL4);

In 2011, ISO/IEC 9075:2000N, SQL:2011 (SQL5);

In 2016, ISO/IEC 9075:2016, SQL: 2016.

SQL standards supported by GBase 8a MPP Cluster

The main features of SQL-92 are supported by default.

5.1.3 data type

GBase 8a MPP Cluster supports the vast majority of data types defined in SQL-92, as well as most of the data types defined in SQL99 and SQL2003.

The data types supported by GBase 8a MPP Cluster are shown in the following table:

Table -53 Data Types

Data types for GBase 8a MPP Cluster	
Numerical type	TINYINT
	SMALLINT
	INT
	BIGINT
	FLOAT
	DOUBLE
	DECIMAL
	NUMERIC
character	CHAR
	VARCHAR
	TEXT
Binary type	BLOB
	BINARY
	VARBINARY
	LONGBLOB
Date and time type	DATE
	DATETIME
	TIME
	TIMESTAMP

Note: There are differences in the scope of the following data types between the gcluster layer and the gnode layer. The support scope of the gcluster layer is larger than that of the gnode layer. It is recommended to use the smaller support scope in application development to facilitate unified processing of applications in the gcluster and gnode layers. The scope of data types in this chapter is uniformly described as the scope supported by gcluster and gnode, that is, the smaller gnode range shall prevail.

data type	Gcluster layer range	Gnode layer range
timestamp	Max 2038-01-19 11:14:07 Min 1970-01-01 08:00:01	Max 2038-01-01 00:59:59 Min 1970-01-01 08:00:01
tinyint	Max 127 Min -128	Max 127 Min -127
smallint	Maximum 32767 Min -32768	Maximum 32767 Min -32767
bigint	Maximum value 9223372036854775807 Min -9223372036854775806	Maximum value 9223372036854775806 Min -9223372036854775806

5.1.3.1 value type

GBase 8a MPP Cluster supports data types including strict numerical data types (TINYINT, SMALLINT, INT, BIGINT, DECIMAL, NUMERIC), as well as approximate numerical data types (FLOAT, DOUBLE).

In order to use storage space more effectively, please try to use the most accurate type. For example, if an integer column is used for values between 1 and 127, TINYINT is the best type.

In order to store a larger range of numerical values, users can choose between BIGINT or DECIMAL types.

The numerical types supported by GBase 8a MPP Cluster are shown in the following table:

Table 54 Numerical Types

Type Name	minimum value	Maximum value	Bytes occupied
TINYINT	-127	one hundred and twenty-seven	one
SMALLINT	-32767	thirty-two thousand seven hundred and sixty-seven	two
INT(INTEGER)	-2147483647	two billion one hundred and forty-seven million four hundred and eighty-three thousand six hundred and	four

Type Name	minimum value	Maximum value	Bytes occupied
		forty-seven	
BIGINT	-9223372036854775806	9223372036854775806	eight
FLOAT	-3.40E+38	3.40E+38	four
DOUBLE	-1.7976931348623157E+308	1.7976931348623157E+308	eight
DECIMAL[(M[, D])]	-(1E+M -1)/(1E+D)	(1E+M -1)/(1E+D)	Dynamic calculation
NUMERIC[(M[, D])]	-(1E+M -1)/(1E+D)	(1E+M -1)/(1E+D)	Dynamic calculation

5.1.3.1.1 TINYINT

Integer type, with a value range of -127 to 127, and TINYINT occupying 1 byte.

5.1.3.1.2 SMALLINT

Integer type. Its value range is -32767 to 32767, and SMALLINT occupies 2 bytes.

5.1.3.1.3 INT

Integer type. Synonym for INTEGER. Its value range is -2147483647 to 2147483647, with INT occupying 4 bytes.

5.1.3.1.4 BIGINT

Integer type. Its value range is -9223372036854775806 to 9223372036854775806, with BIGINT occupying 8 bytes.

Example

Example 1: The defined column data type is BIGINT.

```
CREATE TABLE products(productnum BIGINT);
INSERT INTO products(productnum) VALUES(100);

gbase> SELECT productnum FROM products;
+------------+
| productnum |
+------------+
```

```
|              100 |
+------------+
1 row in set

gbase> DESC products;
+------------+------------+------+-----+---------+-------+
| Field      | Type       | Null | Key | Default | Extra |
+------------+------------+------+-----+---------+-------+
| productnum | bigint(20) | YES  |     | NULL    |       |
+------------+------------+------+-----+---------+-------+
1 row in set (Elapsed: 00:00:00.01)
```

5.1.3.1.5 FLOAT

FLOAT represents a single precision floating-point value that occupies 4 bytes and does not store an accurate value. The allowed values are -3.402823466E+38 to -1.175494351E-38, 0, 1.175494351E-38 to 3.402823466E+38. These are theoretical limitations based on IEEE standards. The actual range may be slightly smaller depending on the hardware or operating system.

GBase 8a MPP Cluster allows the use of bits to specify precision, i.e. FLOAT (X), within parentheses after the keyword FLOAT. The precision from 0 to 24 corresponds to a 4-byte single precision for the FLOAT column, and the precision from 25 to 53 corresponds to an 8-byte double precision for the DOUBLE column. The defined column data type is FLOAT (M), and when the total number of digits is greater than 23, a maximum of 15 decimal places are supported. When 24<=X<=53, FLOAT (X) is equivalent to DOUBLE (X). At the same time, the GBase 8a MPP Cluster allows the use of non-standard syntax FLOAT (M, D) (M is the total number of integer digits and decimal digits, and D is the number of decimals). The GBase 8a MPP Cluster rounds the value when saving it.

Example

Example 1: The defined column data type is FLOAT.

```
CREATE TABLE products(productnum FLOAT);
INSERT INTO products(productnum) VALUES(-19000.44365),
(-19000.48365),(1.44365),(1.443658);

gbase> SELECT productnum FROM products;
+--------------+
| productnum |
+--------------+
```

```
|     -19000.4 |
|     -19000.5 |
|       1.44365 |
|       1.44366 |
+--------------+
4 rows in set
```

Example 2: The defined column data type is FLOAT (M). When the total number of digits is less than or equal to 23, only one significant digit is retained in the decimal part, and the system will automatically round the digits.

```
CREATE TABLE products (a FLOAT(20),b FLOAT(28));
INSERT INTO products (a,b) VALUES(-19000.44365,-19000.44365);

gbase> SELECT * FROM products;
+----------+--------------+
| a              | b                     |
+----------+--------------+
| -19000.4 | -19000.44365 |
+----------+--------------+
1 row in set
```

Example 3: If the defined column data type is FLOAT (20,5) and a precision of 5 is specified, the decimal part is retained to 5 digits.

```
CREATE TABLE products(productnum FLOAT(20,5));
INSERT INTO products(productnum)
VALUES(19000.44365),(19000.443652);

gbase> SELECT productnum FROM products;
+-------------+
| productnum    |
+-------------+
| 19000.44336 |
| 19000.44336 |
+-------------+
2 rows in set
```

Example 4: The defined column data type is FLOAT (7,4), and when the inserted data is 999.00009, its approximate value is 999.0001, which is automatically rounded.

```
CREATE TABLE products(productnum FLOAT(7,4));
INSERT INTO products(productnum) VALUES(999.00009);

gbase> SELECT productnum FROM products;
+------------+
```

```
| productnum |
+------------+
|     999.0001 |
+------------+
1 row in set
```

5.1.3.1.6 DOUBLE

DOUBLE represents a double precision floating-point value that occupies 8 bytes and does not store an accurate value. The allowed values are -1.7976931348623157E+308 to -2.2250738585072014E-308, 0, 2.2250738585072014E-308 to 1.7976931348623157E+308. These are theoretical limitations, based on IEEE standards, and the actual scope may be slightly smaller depending on the hardware or operating system.

GBase 8a MPP Cluster allows the selection of bit specific precision, i.e. DOUBLE (X), within parentheses after the keyword DOUBLE. The precision from 0 to 23 corresponds to a 4-byte single precision for the FLOAT column, and the precision from 24 to 53 corresponds to an 8-byte double precision for the DOUBLE column. When 0<=X<=23, FLOAT (X) is equivalent to DOUBLE (X).

At the same time, the GBase 8a MPP Cluster allows the use of non-standard syntax DOUBLE (M, D) (M is the total number of integer digits and decimal digits, and D is the number of decimals). The GBase 8a MPP Cluster performs rounding when saving values.

Example

Example 1: The defined column data type is DOUBLE.

```
CREATE TABLE products(productnum DOUBLE);
INSERT INTO products(productnum) VALUES(-19000.44365);

gbase> DESC products;
+------------+--------+------+-----+---------+-------+
| Field      | Type   | Null | Key | Default | Extra |
+------------+--------+------+-----+---------+-------+
| productnum | double | YES  |     | NULL    |       |
+------------+--------+------+-----+---------+-------+
1 row in set

gbase> SELECT productnum FROM products;
```

```
+--------------+
| productnum   |
+--------------+
| -19000.44365 |
+--------------+
1 row in set
```

5.1.3.1.7 **DECIMAL**

DECIMAL [(M [, D])] represents an exact value that stores values ranging from -(1E+M-1)/(1E+D) to (1E+M-1)/(1E+D).

In DECIMAL [(MS [, D])] data type, M is the total number of digits, and the maximum supported length is 65; D is the number of digits after the decimal point, and the maximum supported length is 30.

In scenarios where high numerical accuracy is not required, M in DECIMAL can be defined as M ⩽ 18, which can achieve better computational performance.

DECIMAL is used to store data that strictly requires numerical accuracy, such as monetary data, in which case precision needs to be specified:

salary DECIMAL(5,2)

In DECIMAL (5,2), 5 represents the total number of digits (the sum of integer and decimal places), and 2 represents the number of decimal places. The minimum value that can be stored in the sales column is -999.99, and the maximum value is 999.99.

The maximum range of DECIMAL type values is limited by the given precision and decimal range. When exceeding the decimal range, it will be truncated to the set number of decimal places according to the rounding principle.

When defining a DECIMAL type data column, if both M and D are omitted, M takes a value of 10 and D takes a value of 0, which is DECIMAL (10,0). If only the M value is specified and D value is omitted, when inserting a non integer value, it will be rounded to the integer digits according to the rounding principle.


be careful

Comparing decimal with time only supports the comparison of decimal constants with datetime, and does not support the comparison of decimal columns with datetime.

For example, if column g in the table is of type decimal and column f is of type datetime, then

support

select * from t1 where g=cast('20220212112059.010000' as date);

I won't support it

Select * from t1 where g=f;

Example

Example 1: The defined column data type is DECIMAL (18,5).

```
CREATE TABLE products(productnum DECIMAL(18,5));
INSERT INTO products(productnum) VALUES(19000.44365);

gbase> DESC products;
+------------+---------------+------+-----+---------+-------+
| Field      | Type          | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| productnum | decimal(18,5) | YES  |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
1 row in set

gbase> SELECT productnum FROM products;
+-------------+
| productnum  |
+-------------+
| 19000.44365 |
+-------------+
1 row in set
```

Example 2: If the defined column data type is DECIMAL and both M and D are omitted, then the default value for M is 10 and the default value for D is 0.

```
gbase> CREATE TABLE products(productnum DECIMAL);
Query OK, 0 rows affected

gbase> DESC products;
+------------+---------------+------+-----+---------+-------+
```

```
| Field          | Type                | Null | Key | Default | Extra |
+------------+---------------+------+-----+---------+-------+
| productnum | decimal(10,0) | YES  |     | NULL    |       |
+------------+---------------+------+-----+---------+-------+
1 row in set
```

Example 3: Define the column data type as DECIMAL (M, D), and report an error message when the inserted data exceeds the total number of digits M; When the precision D is exceeded, the decimal part is rounded off.

```
gbase> CREATE TABLE products(productnum DECIMAL(8,3));
Query OK, 0 rows affected

gbase> INSERT INTO products(productnum) VALUES(191220.443);
ERROR 1264 (22003): Out of range value for column 'productnum' at row 1

gbase> INSERT INTO products(productnum) VALUES(19122.4436);
Query OK, 1 row affected, 1 warning

gbase> SELECT productnum FROM products;
+------------+
| productnum |
+------------+
|   19122.444 |
+------------+
1 row in set
```

5.1.3.1.8 NUMERIC

The NUMERIC data type is completely equivalent to the DECIMAL data type.

5.1.3.2 Character type

GBase 8a MPP Cluster currently supports four character types, as shown in the table below:

Table 55 Character Types

Type Name	Maximum length
CHAR	255 characters
VARCHAR	32KB 10922 characters (utf8) 16383 characters (gbk) 8191 characters (gb18030)

Type Name	Maximum length
	8191 characters (utf8mb4)
TEXT	64KB 21845(utf8) 32767(gbk) 16383(gb18030) 16383(utf8mb4)
LONGTEXT	64MB 22369621(utf8), 33554432(gbk), 16777216(gb18030) , 16777216(utf8mb4)

5.1.3.2.1 CHAR

CHAR(n)

The CHAR type is only for compatibility with SQL standards, therefore it is not recommended for users to use this data type in actual project application scenarios. It is recommended to use the VARCHAR data type.

CHAR is the abbreviation for CHARACTER. N represents the character length of the string in this column, which ranges from 1 to 255.

When the stored character length is less than the specified length n, fill in with a space on the right side of the string.

When reading the CHAR value, the filled spaces are still retained.

If a string exceeding the maximum length is inserted into a column defined as CHAR type, the system will report an error message.

5.1.3.2.2 VARCHAR

VARCHAR(m)

The storage is a variable length string, with a maximum support of 32KB. m represents the character length of the string in this column, ranging from 1 to 10922 characters for the utf8 character set and 1 to 16383 characters for the gbk character set.

When storing VARCHAR type data, the column definition length is not filled with spaces. When the stored data contains spaces, spaces are retained.

Example

Example 1: The VARCHAR data type does not fill the length of the column definition, but preserves the spaces in the inserted data.

Tables and data used in the example:

```
gbase> CREATE TABLE products (productName VARCHAR(100));
Gbase>Insert INTO products (productName) VALUES ('Nanjing University General Motors');
Gbase>Insert INTO products (productName) VALUES ('Nanjing University General Motors');
gbase> SELECT productName,    LENGTH(productName) AS length,
CHAR_ LENGTH(productName) AS char_ length FROM products;
+----------------+--------+-------------+
| productName        | length | char_ length |
+----------------+--------+-------------+
|Nanda General Motors | 12 | 4|
|Nanjing University General Motors | 14 | 6|
+----------------+--------+-------------+
2 rows in set
Gbase>SELECT productName from products WHERE
productName='Nanjing University General Motors';
+--------------+
| productName    |
+--------------+
|Nanjing University General Motors|
+--------------+
1 row in set
```

Preserve spaces in the original data in the query results:

```
Gbase>SELECT productName from products WHERE
productName='Nanjing University General Motors';
+----------------+
| productName        |
+----------------+
|Nanjing University General Motors|
+----------------+
1 row in set
```

5.1.3.2.3 TEXT

The TEXT type is only for compatibility with other database types, and it is recommended to use the VARCHAR type.

The maximum support for the text type is 64KB. When defining a text column, a

DEFAULT value cannot be specified for it.

5.1.3.2.4 LONGTEXT

The LONGTEXT type supports a maximum of 64MB and cannot be assigned a DEFAULT value.

5.1.3.3 Binary data type

GBase 8a MPP Cluster currently supports the following binary data types, as shown in the table below:

Table -56 Binary Data Types

Type Name	Maximum length (bytes)
BLOB	thirty-two thousand seven hundred and sixty-seven
LONGBLOB	sixty-seven million one hundred and eight thousand eight hundred and sixty-four

Using the BLOB data type has the following constraints:

The BLOB column supports a storage capacity of 32KB.

When creating a table, the BLOB column cannot have a DEFAULT value.

In the query statement, the BLOB column does not support filtering criteria.

The BLAB column in the query statement does not support the OLAP function.

5.1.3.4 Date and time type

GBase 8a MPP Cluster supports date and time types.

Table -57 Date and Time Types

Type Name	minimum value	Maximum value	format
DATE	0001-01-01	9999-12-31	YYYY-MM-dd
DATETIME	0001-01-01 00:00:00.000000	9999-12-31 23:59:59.999999	YYYY-MM-dd HH:MI:SS.ffffff
TIME	-838:59:59	838:59:59	HHH:MI:SS
TIMESTAMP	1970-01-01 08:00:01	2038-01-01 00:59:59	YYYY-MM-DD HH:MI:SS

When using date and time types, users should provide the correct format, such as YYYY-MM-DD, YYYY-MM-DD HH: MI: SS.

5.1.3.4.1 DATE

Date type. The supported range is "0001-01-01" to "9999-12-31".

GBase 8a MPP Cluster displays DATE values in the "YYYY-MM-DD" format.

Example

Example 1: Insert a standard DATE value.

```
gbase> CREATE TABLE products (productDate DATE);
Query OK, 0 rows affected

gbase> INSERT INTO products(productDate) VALUES('2010-09-01');
Query OK, 1 row affected

gbase> SELECT productDate FROM products;
+-------------+
| productDate |
+-------------+
| 2010-09-01  |
+-------------+
1 row in set
```

Example 2: Insert a NULL value.

```
gbase> CREATE TABLE products (productDate DATE);
Query OK, 0 rows affected

gbase> INSERT INTO products(productDate) VALUES(NULL);
Query OK, 1 row affected

gbase> SELECT productDate FROM products;
+-------------+
| productDate |
+-------------+
| NULL        |
+-------------+
1 rows in set
```

Example 3: Insert an illegal DATE value, and the system reports an error message.

```
gbase> CREATE TABLE products (productDate DATE);
```

```
Query OK, 0 rows affected

gbase> INSERT INTO products(productDate) VALUES('2010-09-31');

ERROR 1292 (22007): Incorrect date value: '2010-09-31' for column 'productDate'
at row 1
```

5.1.3.4.2 TIME

GBase 8a MPP Cluster retrieves and displays the TIME value in the "HH: MM: SS" format (or "HHH: MM: SS" format), which is a string.

The range of TIME is "-838:59:59" to "838:59:59". The TIME type can be used not only to represent the time of the day, but also to represent the elapsed time or the time interval between two events (which may be much larger than 24 hours or a negative value).

For a TIME value specified in a string that contains a time delimiter, hours, minutes, or seconds less than 10 may not be specified as a two digit numerical value. 8:3:2 is consistent with 08:03:02.

The TIME value can be specified in multiple formats:

A string in the format of 'D HH: MM: SS. fraction'. Any "loose" syntax shown below can be used: 'HH: MM: SS. fraction', 'HH: MM: SS',' HH: MM ',' D HH: MM: SS', 'D HH: MM', 'D HH', or 'SS'. The D here is a number of days between 0 and 34. Note that the fraction section can be precise to microseconds.

Example

Example 1: Insert a standard TIME value.

```
gbase> CREATE TABLE products (producttime TIME);
Query OK, 0 rows affected

gbase> INSERT INTO products(producttime) VALUES('12:35:23');
Query OK, 1 row affected

gbase> SELECT producttime FROM products;
+-------------+
| producttime |
+-------------+
| 12:35:23    |
+-------------+
1 row in set
```

Example 2: Insert a NULL value.

```
gbase> CREATE TABLE products (producttime TIME);
Query OK, 0 rows affected
gbase> INSERT INTO products(producttime) VALUES(NULL);
Query OK, 1 row affected

gbase> SELECT producttime FROM products;
+-------------+
| producttime |
```

```
+-------------+
| NULL        |
+-------------+
1 rows in set
```

Example 3: Insert an illegal TIME value, and the system reports an error message.

```
gbase> CREATE TABLE products (producttime TIME);
Query OK, 0 rows affected

gbase> INSERT INTO products(producttime) VALUES('14:08:89');
ERROR 1292 (22007): Incorrect time value: '14:08:89' for column 'producttime' at
row 1
```

5.1.3.4.3 DATETIME

GBase 8a MPP Cluster displays DATETIME values in the format "YYYY-MM-DD HH: MI: SS. fraction". Among them, 'fraction' represents microsecond format and supports up to 6 digits.

The combination type of date and time. The supported range is "0001-01-01 00:00:00.00000" to "9999-12-31 23:59:59.999999".

Example

Example 1: Insert a valid DATETIME value.

```
gbase> CREATE TABLE products (productDate DATETIME);
Query OK, 0 rows affected

gbase> INSERT INTO products(productDate) VALUES('2010-09-01
12:09:44');
Query OK, 1 row affected

gbase> SELECT productDate FROM products;
+---------------------+
| productDate         |
+---------------------+
| 2010-09-01 12:09:44 |
+---------------------+
1 row in set
```

Example 2: Insert the current DATETIME value of the system.

```
gbase> INSERT INTO products(productDate) VALUES(NOW());
Query OK, 1 row affected

gbase> SELECT productDate FROM products;
+---------------------+
| productDate         |
+---------------------+
| 2013-10-16 17:51:38 |
+---------------------+
1 row in set
```

Example 3: Insert a NULL value.

```
gbase> INSERT INTO products(productDate) VALUES(NULL);
Query OK, 1 row affected

gbase> SELECT productDate FROM products;
+-------------+
| productDate |
+-------------+
| NULL        |
+-------------+
1 row in set
```

Example 4: Insert a DATETIME value with microseconds.

```
gbase> INSERT INTO products(productDate) VALUES('2013-09-15
12:09:44.123456');
Query OK, 1 row affected

gbase> SELECT productDate FROM products;
+----------------------------+
| productDate                |
+----------------------------+
| 2013-09-15 12:09:44.123456 |
+----------------------------+
1 row in set
```

Example 5: If an illegal DATETIME value is inserted, the system will report an error message.

```
gbase> INSERT INTO products(productDate) VALUES('2010-09-31
12:09:44');
ERROR 1292 (22007): Incorrect datetime value: '2010-09-31 12:09:44' for column
'productDate' at row 1
```

5.1.3.4.4 TIMESTAMP

The TIMESTAMP type is only for compatibility with SQL standards and is not recommended. It is recommended to use the DATETIME data type.

The format of TIMESTAMP is "YYYY-MM-DD HH: MI: SS", and the supported range is "1970-01-01 08:00:01" to "2038-01-01 00:59:59".

```
gbase> CREATE TABLE t (a int,b timestamp DEFAULT
CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP, c
timestamp DEFAULT '2013-01-01 00:00:01');
Query OK, 0 rows affected

gbase> SHOW CREATE TABLE t;
+-------+-----------------------------------------------------------------+
| Table | Create Table
|
+-------+-----------------------------------------------------------------+
| t     | CREATE TABLE "t" (
  "a" int(11) DEFAULT NULL,
  "b" timestamp NOT NULL DEFAULT CURRENT_ TIMESTAMP ON
UPDATE CURRENT_ TIMESTAMP,
  "c" timestamp NOT NULL DEFAULT '2013-01-01 00:00:01'
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
```

```
tablespace'               |
+-------+-----------------------------------------------------------------+
1 row in set
```

TIMESTAMP usage restrictions

The following limitations apply to scenarios where TIMESTAMP data columns are automatically updated:

Using DEFAULT Current_ TIMESTAMP ON UPDATE CURRENT_ After the TIMESTAMP attribute, the TIMESTAMP column automatically updates its value when it supports Insert, UPDATE, and MERGE. When creating a table, one or more TIMESTAMP columns can be defined in a table. If only one is defined, DEFAULT Current_ TIMESTAMP ON UPDATE CURRENT_ The TIMESTAMP attribute can be left unspecified during creation and will be automatically filled in by the system. If multiple columns need to be defined, the first TIMESTAMP column must specify DEFAULT Current_ TIMESTAMP ON UPDATE CURRENT_ TIMESTAMP attribute, and other TIMESTAMP columns cannot specify DEFAULT Current_ TIMESTAMP ON UPDATE CURRENT_ TIMESTAMP attribute.

5.1.4 constant

5.1.4.1 character string

A string is a sequence of multiple characters enclosed by a single quotation mark ".

For example: 'a string'.

Multiple quoted strings connected together are equivalent to a single string. The following two lines are written in the same way:

'a string'

'a' ' ' 'string'

In a string, the determined sequence has a special meaning. Each sequence starts with the backslash symbol "", which is called the escape character. GBase 8a MPP Cluster supports the following escape character:

Table -58 escape character of String

Escape character	Description
\0	ASCII 0 (NUL) character.
\'	ASCII 39 single quote " character.
\"	ASCII 34 double quotation mark "" character.
\b	ASCII 8 backspace character.
\n	ASCII 10 line breaks.
\r	ASCII 13 carriage return.

Escape character	Description
\t	ASCII 9 tab (TAB).
\\	ASCII 92 backslash "" character.

These symbols are case sensitive. For example, " b" is interpreted as a backspace, but " B" is interpreted as "B".

In all other escape character, backslash is ignored. In other words, backslashes are used to interpret escape character rather than being escaped.

When a string contains quotation marks:

1）A string is referenced with a single quotation mark ", and the single quotation mark " character in the string can be escaped using two single quotation marks' ". Users can also continue to escape by adding a escape character "" before the quotation marks.

2）A string is referenced with a single quotation mark ", and the double quotation mark " in the string does not require special treatment and does not need to be repeatedly specified or escaped.

Example

Example 1: Enclosing a string with single quotation marks' '.

```
gbase> SELECT 'hello', '"hello"', '""hello""', 'hel''lo', '\'hello' FROM dual;
+-------+---------+-----------+--------+--------+
| hello | "hello" | ""hello"" | hel'lo | 'hello |
+-------+---------+-----------+--------+--------+
| hello | "hello" | ""hello"" | hel'lo | 'hello |
+-------+---------+-----------+--------+--------+
1 row in set
```

Example 2: The escape character "" exists in the string.

```
gbase> SELECT 'This\nIs\nFour\nLines' FROM dual;
+--------------------+
| This
Is
Four
Lines |
+--------------------+
| This
Is
Four
Lines |
+--------------------+
1 row in set
```

Example 3: When there is no escape meaning, the backslash symbol is ignored.

```
gbase> SELECT 'disappearing\ backslash' FROM dual;
+------------------------+
| disappearing backslash |
+------------------------+
| disappearing backslash |
+------------------------+
1 row in set
```

If you want to insert binary data into the BLOB field, the following characters must be represented by escape character:

Table -59 escape character

Character	Description
NUL	NUL byte （ASCII 0）。 It needs to be represented by ' 0' (a backslash and an ASCII '0' character).
\	Backslash (ASCII 92). It needs to be represented by ' '.
'	Single quotation mark (ASCII 39). It needs to be represented by ''.
"	Double quotation marks (ASCII 34). It needs to be represented by ''.

Example 4: The type of productBlob field in the created table is BLOB, and there are escape character in the inserted data.

```
gbase> DROP TABLE IF EXISTS products;
Query OK, 0 rows affected

gbase> CREATE TABLE products (productBlob    BLOB);
Query OK, 0 rows affected

gbase> INSERT INTO products
VALUES('abdcdrf\\ghi\'jklm\"nopqrs\0tuvwxyz');
Query OK, 1 row affected

gbase> SELECT productBlob FROM products;
+---------------------------------+
| productBlob                     |
+---------------------------------+
| abdcdrf\ghi'jklm"nopqrs tuvwxyz |
+---------------------------------+
1 row in set
```

When writing code, any string may contain these special characters, so these characters must be escaped before being passed as data in SQL statements to GBase 8a MPP Cluster.

5.1.4.2 number

An integer is represented as a sequence of numbers. Floating point numbers use "." as the separator of a decimal number. These two numerical types can be prefixed with "-" to represent a negative value.

Examples of valid integers:

1221、0、-32

Examples of valid floating point numbers:

-32032.6809E+10、148.00E+13

5.1.4.3 Hexadecimal value

GBase 8a MPP Cluster supports hexadecimal values. In the context of numbers, they are used as numerical equivalents to integers.

In the context of a string, they are treated as a string, and each set of hexadecimal digits is interpreted as characters corresponding to the ASCII code.

Example

Example 1: 0xa is equivalent to the integer 10.

```
gbase> SELECT 0xa+1 FROM dual;
+-------+
| 0xa+1 |
+-------+
|    11 |
+-------+
1 row in set
```

Example 2: Convert "4742617365" to the corresponding ASCII code.

```
gbase> SELECT x'4742617365' FROM dual;
+---------------+
| x'4742617365' |
+---------------+
| GBase         |
+---------------+
1 row in set
```

Example 3: Convert "5061756c" to the corresponding ASCII code.

```
gbase> SELECT 0x5061756c FROM dual;
+------------+
| 0x5061756c |
+------------+
| Paul       |
+------------+
1 row in set
```

The expression 'x'hexstring' is based on standard SQL, while the expression 0x is based on ODBC. The two are equivalent.

Example 4: The HEX() function can be used to convert a string or numerical value into a string in hexadecimal format.

```
gbase> SELECT HEX('cat') FROM dual;
+------------+
| HEX('cat') |
+------------+
| 636174     |
+------------+
1 row in set

gbase> SELECT 0x636174 FROM dual;
+----------+
| 0x636174 |
+----------+
| cat      |
+----------+
1 row in set
```

5.1.4.4 Boolean value

The constant TRUE is equivalent to 1, while the constant FALSE is equivalent to 0.

The name of a constant is not case sensitive.

Example

Example 1: Query the values corresponding to TRUE and FALSE.

```
gbase> SELECT TRUE, true, FALSE, false FROM dual;
+------+------+-------+-------+
| TRUE | TRUE | FALSE | FALSE |
+------+------+-------+-------+
|    1 |    1 |     0 |     0 |
+------+------+-------+-------+
1 row in set
```

5.1.4.5 NULL value

NULL is not case sensitive.

be careful

A NULL value is different from a numeric type of 0 or an empty string of string type.

5.1.5 Functions and Operators

5.1.5.1 Operator

5.1.5.1.1 Operator precedence

explain

The operator priority is listed below, from highest to lowest.

Operators on the same line have the same priority.

```
BINARY, COLLATE
!
-(unary minus), ~ (unary bit inversion)
^
*,/,DIV,%,MOD
-,+
<<,>>
&
|
=,<=>,>=,>,<=,<,<>,!=, IS,LIKE,REGEXP,IN
BETWEEN,CASE,WHEN,THEN,ELSE
NOT
&&,AND
OR,XOR
:=
```

5.1.5.1.2 Parentheses

explain

```
(...)
```

Parentheses are used to specify the order of operations for an expression, and operators placed in parentheses are executed first.

Example

Example 1: Without parentheses, the expression performs multiplication before addition.

```
gbase> SELECT 1+2*3 FROM dual;
+-------+
| 1+2*3 |
+-------+
|     7 |
+-------+
1 row in set
```

Example 2: Using parentheses, the expression first performs the addition operation inside the parentheses, and then performs the multiplication operation outside the parentheses.

```
gbase> SELECT (1+2)*3 FROM dual;
+---------+
| (1+2)*3 |
+---------+
|       9 |
+---------+
1 row in set
```

5.1.5.1.3 Compare functions and operators

summary

The result of a comparison operation is 1 (TRUE), 0 (FALSE), or NULL.

These operations can be used on numbers and strings. As needed, strings will be automatically converted to numbers, and numbers can also be automatically converted to strings.

explain

Although the values obtained by some functions in this chapter, such as GREATEST() and LEAST(), do not include 1 (TRUE), 0 (FALSE), or NULL, when comparing parameter values, they are also based on the following rules.

- GBase 8a MPP Cluster uses the following rules for comparison:

- If one or two parameters are NULL, the comparison result is NULL, except for the<=>comparison operator (with a NULL parameter, the comparison result of<=>is not NULL).
- If both parameters are strings in a comparison operation, they are compared as strings (default case insensitive).
- If both parameters are numerical values, they are compared by value.
- If one parameter in the comparison operation is a string and the other is a numerical value, convert the string to a numerical value and compare according to the numerical value. Convert a string to a numerical value. For strings that start with a number, the result of converting to a numerical value is to truncate the previous numerical part; For strings that do not start with a number, the result of converting to a numerical value is 0.
- If a hexadecimal value is not compared to a number, it will be treated as a binary string.
- If one of the parameters is a DATETIME column type and the other parameters are a constant, this constant is converted to a timestamp before the comparison is executed.
- In other cases, parameters are compared as floating-point (REAL) numbers.

 **be careful**

The parameter in IN() is not like this. For safety reasons, it is recommended that users use the complete DATETIME/DATE/TIME string when comparing.

For comparison purposes, users can use the CAST() function to convert a value to another type.

5.1.5.1.3.1 = 等于

grammar

```
a=b
```

Expression Description

If two operands are equal, returns 1.

Example

Example 1: Both operands are numbers.

```
gbase> SELECT 1 = 0 FROM dual;
+-------+
| 1 = 0 |
+-------+
|     0 |
+-------+
1 row in set
```

Example 2: Comparing numbers with strings.

```
gbase> SELECT '0' = 0 FROM dual;
+---------+
| '0' = 0 |
+---------+
|       1 |
+---------+
1 row in set
```

Example 3: Compare a floating point number with a number.

```
gbase> SELECT 0.0 = 0 FROM dual;
+---------+
| 0.0 = 0 |
+---------+
|       1 |
+---------+
1 row in set
```

Example 4: Both operands are NULL.

```
gbase> SELECT NULL = NULL FROM dual;
+--------------+
| NULL = NULL    |
+--------------+
|                NULL |
+--------------+
1 row in set
```

5.1.5.1.3.2 <=>NULL value is safe to be equal to

grammar

```
a<=>b
```

Expression Description

This operator performs equality comparison like the "=" operator, but the results obtained are different from "=" in the following two cases:

1. If all operands are NULL, then 1 is returned instead of NULL.
2. If there is only one operand that is NULL, then 0 is returned instead of NULL.

Example

Example 1: All operands are NULL, or some operands are NULL.

```
gbase> SELECT 1 <=> 1, NULL <=> NULL, 1 <=> NULL FROM dual;
+---------+---------------+-------------+
| 1 <=> 1 | NULL <=> NULL | 1 <=> NULL    |
+---------+---------------+-------------+
|             1 |                         1 |                     0 |
+---------+---------------+-------------+
1 row in set
```

Example 2: Result of the=operator

```
gbase> SELECT 1 = 1, NULL = NULL, 1 = NULL FROM dual;
+-------+-------------+-----------+
| 1 = 1 | NULL = NULL | 1 = NULL  |
+-------+-------------+-----------+
|     1 |        NULL |      NULL |
+-------+-------------+-----------+
1 row in set
```

5.1.5.1.3.3 <> , != Not equal to

grammar

```
A<>b or a= b
```

Expression Description

If two operands are not equal, returns 1. If one of the operands is NULL, it returns NULL.

Example

Example 1: Operands are all strings.

```
gbase> SELECT '01'<>'1' FROM dual;
+-----------+
| '01'<>'1' |
+-----------+
|         1 |
+-----------+
1 row in set
```

Example 2: One of the operands is a string.

```
gbase> SELECT 01<>'1' FROM dual;
+---------+
| 01<>'1' |
+---------+
|       0 |
+---------+
1 row in set
```

Example 3: Operands are all strings.

```
gbase> SELECT 'zapp' <> 'zappp' FROM dual;
+--------------------+
| 'zapp' <> 'zappp' |
+--------------------+
|                              1 |
+--------------------+
1 row in set
```

5.1.5.1.3.4 <= 小于或者等于

grammar

```
a<=b
```

Expression Description

If a is less than or equal to b, it returns 1. If one of the operands is NULL, it returns NULL.

Example

Example 1: Both operands are numbers.

```
gbase> SELECT 0.1 <= 2 FROM dual;
+----------+
| 0.1 <= 2 |
+----------+
|               1 |
+----------+
1 row in set
```

5.1.5.1.3.5 <Less than

grammar

```
a<b
```

Expression Description

If a is less than b, it returns 1. If one of the operands is NULL, it returns NULL.

Example

Example 1: Both operands are numbers.

```
gbase> SELECT 2 < 2 FROM dual;
+-------+
| 2 < 2 |
+-------+
|     0 |
+-------+
1 row in set
```

5.1.5.1.3.6 >= 大于或者等于

grammar

```
a>=b
```

Expression Description

If a is greater than or equal to b, it returns 1. If one of the operands is NULL, it returns NULL.

Example

Example 1: Both operands are numbers.

```
gbase> SELECT 2 >= 2 FROM dual;
+--------+
| 2 >= 2 |
+--------+
|      1 |
+--------+
1 row in set
```

5.1.5.1.3.7 >Greater than

grammar

```
a>b
```

Expression Description

If a is greater than b, it returns 1. If one of the operands is NULL, it returns NULL.

Example

Example 1: Both operands are numbers.

```
gbase> SELECT 2 > 2 FROM dual;
+-------+
| 2 > 2 |
+-------+
|     0 |
+-------+
1 row in set
```

5.1.5.1.3.8 IS [NOT]

grammar

```
IS [NOT] <NULL|TRUE|FALSE|UNKNOWN>
```

Function Description

Test a value based on a Boolean value, which can be NULL, TRUE, FALSE, or UNKNOWN.

Example

Example 1: Using IS statements to verify 1, 0, and NULL.

```
gbase> SELECT 1 IS TRUE, 0 IS FALSE, NULL IS UNKNOWN FROM
dual;
+-----------+------------+------------------+
| 1 IS TRUE | 0 IS FALSE | NULL IS UNKNOWN  |
+-----------+------------+------------------+
|         1 |          1 |                1 |
+-----------+------------+------------------+
1 row in set
```

Example 2: Using the IS NOT statement to verify 1, 0, and NULL.

```
gbase> SELECT 1 IS NOT UNKNOWN, 0 IS NOT UNKNOWN, NULL IS
NOT UNKNOWN FROM dual;
+-----------------+-----------------+---------------------+
| 1 IS NOT UNKNOWN | 0 IS NOT UNKNOWN | NULL IS NOT UNKNOWN
|
+-----------------+-----------------+---------------------+
|                1 |                1 |                   0 |
+-----------------+-----------------+---------------------+
1 row in set
```

Example 3: Using an IS statement to verify whether a value is NULL.

```
gbase> SELECT 1 IS NULL, 0 IS NULL, NULL IS NULL FROM dual;
+-----------+-----------+--------------+
| 1 IS NULL | 0 IS NULL | NULL IS NULL  |
+-----------+-----------+--------------+
|         0 |         0 |            1 |
+-----------+-----------+--------------+
1 row in set
```

Example 4: Using the IS NOT statement to verify whether a value is NULL.

```
gbase> SELECT 1 IS NOT NULL, 0 IS NOT NULL, NULL IS NOT NULL
FROM dual;
+---------------+---------------+------------------+
| 1 IS NOT NULL | 0 IS NOT NULL | NULL IS NOT NULL  |
+---------------+---------------+------------------+
|             1 |             1 |                0 |
+---------------+---------------+------------------+
1 row in set
```

5.1.5.1.3.9 expr BETWEEN min AND max

grammar

```
expr BETWEEN min AND max
```

Function Description

If the value of expr is between min and max (including min and max), return 1; otherwise, return 0.

If all parameters are of the same type, the above relationship is equivalent to an expression (min<=expr AND expr<=max). Other types of conversions are carried out according to the rules mentioned at the beginning of this chapter, and are applicable to any of the three parameters.

Example

Example 1: All parameters are of the same type, expr is not in min and max.

```
gbase> SELECT 1 BETWEEN 2 AND 3 FROM dual;
+-------------------+
| 1 BETWEEN 2 AND 3 |
+-------------------+
|                 0 |
+-------------------+
1 row in set
```

Example 2: All parameters are of the same type, expr in min and max.

```
gbase> SELECT 'b' BETWEEN 'a' AND 'c' FROM dual;
+-------------------------+
| 'b' BETWEEN 'a' AND 'c' |
+-------------------------+
|                       1 |
+-------------------------+
1 row in set
```

Example 3: Parameters contain numbers and strings.

```
gbase> SELECT 2 BETWEEN 2 AND '3' FROM dual;
+---------------------+
| 2 BETWEEN 2 AND '3' |
+---------------------+
|                   1 |
+---------------------+
1 row in set
```

5.1.5.1.3.10 expr NOT BETWEEN min AND max

grammar

```
expr NOT BETWEEN min AND max
```

Equivalent to

```
NOT(expr BETWEEN min AND max)
```

5.1.5.1.3.11 COALESCE(value,...)

grammar

```
COALESCE(value,...)
```

Function Description

The return value is the first non NULL value in the list, and in the case of all NULL values, the return value is NULL.

Example

Example 1: One of the parameter values is NULL.

```
gbase> SELECT COALESCE(NULL,1) FROM dual;
+------------------+
| COALESCE(NULL,1) |
+------------------+
|                1 |
+------------------+
1 row in set
```

Example 2: Each parameter value is NULL.

```
gbase> SELECT COALESCE(NULL,NULL,NULL) FROM dual;
+--------------------------+
| COALESCE(NULL,NULL,NULL) |
+--------------------------+
|                     NULL |
+--------------------------+
1 row in set
```

5.1.5.1.3.12 GREATEST(value1,value2,...)

grammar

```
GREATEST(value1,value2,...)
```

Function Description

When there are two or more parameters, the return value is the maximum parameter value.

When one of the parameters is NULL, it directly returns NULL.

When the parameters are all strings, the default is case insensitive. If you want to perform a case sensitive comparison of string values, add BINARY before the sensitive string parameters.

- These parameters are compared using the following rules:
  - If the return value is in the INTEGER context or if all parameters are integer values, they are compared using integers;
  - If the return value is in the REAL context or all parameters are real values, then they are compared using real numbers;
  - If all parameters are case sensitive strings, then parameter comparison is also case sensitive; In other cases, parameters are relatively case insensitive.

Example

Example 1: The parameter value is an integer number.

```
gbase> SELECT GREATEST(2,0) FROM dual;
+---------------+
| GREATEST(2,0) |
+---------------+
|             2 |
+---------------+
1 row in set
```

Example 2: The parameter value is a floating point number.

```
gbase> SELECT GREATEST(34.0,3.0,5.0,767.0) FROM dual;
+------------------------------+
| GREATEST(34.0,3.0,5.0,767.0) |
+------------------------------+
|                        767.0 |
+------------------------------+
1 row in set
```

Example 3: The parameter value is a string and is not case sensitive.

```
gbase> SELECT GREATEST('B','a','C') FROM dual;
+----------------------+
| GREATEST('B','a','C') |
+----------------------+
| C                    |
+----------------------+
1 row in set
```

Example 4: The parameter value is a string, with BINARY added before the string parameter, which is case sensitive.

```
gbase> SELECT GREATEST('B',BINARY 'a','C') FROM dual;
+------------------------------+
| GREATEST('B',BINARY 'a','C') |
+------------------------------+
| a                            |
+------------------------------+
1 row in set
```

Example 5: If the parameter value contains NULL, the execution result is NULL.

```
gbase> SELECT GREATEST('B',NULL,'C') FROM dual;
```

```
+------------------------+
| GREATEST('B',NULL,'C') |
+------------------------+
| NULL                   |
+------------------------+
1 row in set
```

5.1.5.1.3.13 expr IN (value,...)

grammar

```
expr IN (value,...)
```

Function Description

If expr is any value in the IN list, it will return 1, otherwise it will return 0.

If all values in the IN list are constants, then all values are calculated and sorted according to the type of expr.

If the expression on the left is NULL, or if no matching value is found in the list and one of the expressions in the list is NULL, IN returns NULL.

The IN() syntax can also be used for sub query types.

Example

Example 1: expr is not any value in the IN list.

```
gbase> SELECT 2 IN (0,3,5,'8') FROM dual;
+------------------+
| 2 IN (0,3,5,'8') |
+------------------+
|                0 |
+------------------+
1 row in set
```

Example 2: expr is a value in the IN list.

```
gbase> SELECT '1' IN (0,3,5,'1') as v_ 1,'1' IN (0,3,5,NULL) as v_ null FROM
```

```
dual;
+-----+--------+
| v_ 1 | v_ null |
+-----+--------+
|     1 |      NULL |
+-----+--------+
1 row in set
```

Example 3: The value of expr is NULL.

```
gbase> SELECT NULL IN (0,3,5,'wefwf') FROM dual;
+-------------------------+
| NULL IN (0,3,5,'wefwf') |
+-------------------------+
|                            NULL |
+-------------------------+
1 row in set
```

Example 4: The subquery contains the IN() function.

Tables and data used in the example:

```
CREATE TABLE sc (sno VARCHAR(4), grade INT);
INSERT INTO sc VALUES ('101',82),('102',59),('103',90),('104',88),('106',82);
```

Search for student IDs of students who have passed all courses.

```
gbase> SELECT sno FROM sc WHERE grade IN (SELECT grade FROM sc
WHERE grade>60) GROUP BY sno;
+------+
| sno    |
+------+
| 103    |
| 101    |
```

```
| 104    |
| 106    |
+------+
4 rows in set
```

5.1.5.1.3.14 expr NOT IN (value,...)

grammar

```
expr NOT IN (value,...)
```

Equivalent to

```
NOT(expr IN (value,...))
```

5.1.5.1.3.15 ISNULL(expr)

grammar

```
ISNULL(expr)
```

Function Description

If expr is NULL, the return value of ISNULL() is 1, otherwise the return value is 0.

Example

Example 1: The value of expr is not NULL.

```
gbase> SELECT ISNULL(1+1) FROM dual;
+-------------+
| ISNULL(1+1) |
+-------------+
|                   0 |
+-------------+
1 row in set
```

Example 2: The result of 1/0 is NULL, and the return value of ISNULL() is 1.

```
gbase> SELECT ISNULL(1/0) FROM dual;
```

```
+-------------+
| ISNULL(1/0) |
+-------------+
|           1 |
+-------------+
1 row in set
```

Example 3: Comparing NULL values with '=' results in ISNULL() being 1.

```
gbase> SELECT ISNULL(NULL=NULL) FROM dual;
+-------------------+
| ISNULL(NULL=NULL) |
+-------------------+
|                 1 |
+-------------------+
1 row in set
```

The ISNULL() function shares some of the same characteristics as the IS NULL comparison operator. Please refer to Example 3 in "5.1.5.1.3.8 IS [NOT]" for the use of IS NULL.

5.1.5.1.3.16 LEAST(value1,value2,...)

grammar

```
LEAST(value1,value2,...)
```

Function Description

There are two or more parameters, returning the smallest parameter value. If any variable is NULL, the return value of LEAST() is NULL.

LEAST() compares parameters based on the same rules as GREATEST().

Example

Example 1: If the parameter value is an integer number, the smallest parameter value is returned.

```
gbase> SELECT LEAST(2,0) FROM dual;
+------------+
| LEAST(2,0) |
+------------+
|          0 |
+------------+
1 row in set
```

Example 2: If the parameter value is a floating-point number, the smallest parameter value is returned.

```
gbase> SELECT LEAST(34.0,3.0,5.0,767.0) FROM dual;
+--------------------------+
| LEAST(34.0,3.0,5.0,767.0) |
+--------------------------+
|                      3.0 |
+--------------------------+
1 row in set
```

Example 3: The parameter value is a string and is not case sensitive.

```
gbase> SELECT LEAST('B','A','C') FROM dual;
+--------------------+
| LEAST('B','A','C') |
+--------------------+
| A                  |
+--------------------+
1 row in set

gbase> SELECT LEAST('B','a', 'C') FROM dual;
+---------------------+
| LEAST('B','a', 'C') |
+---------------------+
| a                   |
+---------------------+
1 row in set
```

Example 4: The parameter value is a string, with BINARY added before the string parameter, which is case sensitive.

```
gbase> SELECT LEAST(BINARY 'B',BINARY 'a', 'C') FROM dual;
+-----------------------------------+
| LEAST(BINARY 'B',BINARY 'a', 'C') |
+-----------------------------------+
| B                                 |
+-----------------------------------+
1 row in set
```

Example 5: If the parameter value contains NULL, the execution result is NULL.

```
gbase> SELECT LEAST('C',NULL,'B') FROM dual;
```

```
+---------------------+
| LEAST('C',NULL,'B') |
+---------------------+
| NULL                |
+---------------------+
1 row in set
```

5.1.5.1.4 Logical operator

summary

In SQL, all logical operators return values of TRUE, FALSE, or NULL (UNKNOWN), which are represented by 1 (TRUE), 0 (FALSE), and NULL.

5.1.5.1.4.1 NOT，！Logical non

Operator Description

If the operand is 0, return 1; If the operand is non zero, return 0; If the operand is NULL, return NULL.

Example

Example 1: If the operand is non zero, the return value is 0.

```
gbase> SELECT NOT 10 FROM dual;
+--------+
| NOT 10 |
+--------+
|      0 |
+--------+
1 row in set
```

Example 2: The operand is 0 and the return value is 1.

```
gbase> SELECT NOT 0 FROM dual;
+-------+
| NOT 0 |
+-------+
|     1 |
+-------+
1 row in set
```

Example 3: If the operand is NULL, the return value is NULL.

```
gbase> SELECT NOT NULL FROM dual;
+-----------+
| NOT NULL  |
+-----------+
|      NULL |
+-----------+
1 row in set
```

Example 4: The value of the expression is non zero, and the return value is 0.

```
gbase> SELECT ! (1+1) FROM dual;
+---------+
| ! (1+1) |
+---------+
|       0 |
+---------+
1 row in set
```

Example 5: Expression! 1+1 is equivalent to (! 1)+1, and the execution result is 1.

```
gbase> SELECT ! 1+1 FROM dual;
+-------+
```

```
| ! 1+1 |
+-------+
|     1 |
+-------+
1 row in set

gbase> SELECT (!1)+1 FROM dual;
+--------+
| (!1)+1 |
+--------+
|      1 |
+--------+
1 row in set
```

Example 6: NOT IN...

```
gbase> SELECT 1 NOT IN (2,3,null) FROM dual;
+---------------------+
| 1 NOT IN (2,3,null) |
+---------------------+
|                NULL |
+---------------------+
1 row in set
```

5.1.5.1.4.2 XOR logical XOR

grammar

```
A XOR b is equivalent to (a AND (NOT b)) OR ((NOT a) AND b)
```

Operator Description

When any operand is NULL, the return value is NULL.

For non NULL operands:

XOR	True	False
True	false	really
False	really	false

That is to say, if two values are not the same, the XOR result is true, and vice versa, it is false.

Example

Example 1: If the operand is not NULL, true exclusive or true, and the result is false, the return value is 0.

```
gbase> SELECT 1 XOR 1 FROM dual;
+---------+
| 1 XOR 1 |
+---------+
|       0 |
+---------+
1 row in set
```

Example 2: If the operand is not NULL, true exclusive or false, and the result is true, the return value is 1.

```
gbase> SELECT 1 XOR 0 FROM dual;
+---------+
| 1 XOR 0 |
+---------+
|       1 |
+---------+
1 row in set
```

Example 3: If any operand is NULL, the result is NULL.

```
gbase> SELECT 1 XOR NULL FROM dual;
+-------------+
| 1 XOR NULL  |
```

```
+-------------+
|             NULL |
+-------------+
1 row in set

gbase> SELECT 0 XOR NULL FROM dual;
+-------------+
| 0 XOR NULL    |
+-------------+
|             NULL |
+-------------+
1 row in set
```

Example 4: a XOR b is equivalent to (a AND (NOT b)) OR ((NOT a) AND b).

```
gbase> SELECT 1 XOR 0 FROM dual;
+---------+
| 1 XOR 0 |
+---------+
|           1 |
+---------+
1 row in set

gbase> SELECT (1 AND (NOT 0)) OR ((NOT 1) AND 0) ;
+-----------------------------------+
| (1 AND (NOT 0)) OR ((NOT 1) AND 0) |
+-----------------------------------+
|                                   1 |
+-----------------------------------+
```

```
1 row in set
```

Example 5: Comparing the result of the same number XOR with that number again, the result is 1.

```
gbase> SELECT 1 XOR 1 XOR 1 FROM dual;
+---------------+
| 1 XOR 1 XOR 1 |
+---------------+
|             1 |
+---------------+
1 row in set
```

5.1.5.1.5 Conversion operators and functions

5.1.5.1.5.1 BINARY

Operator Description

Using the BINARY operator before a string allows for case-sensitive comparison of parameter values.

Example

Example 1: BINARY is not used before a string, and it is not case sensitive.

```
gbase> SELECT 'a' = 'A' FROM dual;
+-----------+
| 'a' = 'A' |
+-----------+
|         1 |
+-----------+
1 row in set
```

```
gbase> SELECT 'a' = 'a ' FROM dual;
+------------+
| 'a' = 'a ' |
+------------+
|          1 |
+------------+
1 row in set
```

Example 2: Using BINARY before a string is more case sensitive.

```
gbase> SELECT BINARY 'a' = 'A' FROM dual;
+------------------+
| BINARY 'a' = 'A' |
+------------------+
|                0 |
+------------------+
1 row in set
```

Example 3: Using BINARY before a string to compare trailing spaces.

```
gbase> SELECT BINARY 'a' = 'a ' FROM dual;
+-------------------+
| BINARY 'a' = 'a ' |
+-------------------+
|                 0 |
+-------------------+
1 row in set
```

5.1.5.1.5.2 CAST and CONVERT functions

grammar

```
CAST(expr AS type) , CONVERT(expr,type) , CONVERT(expr USING transcoding_name)
```

Function Description

The CAST() and CONVERT() functions are used to convert a numerical value of one type to another.

- Type can be one of the following values:
  - CHAR、DATE、DATETIME、DECIMAL、TIME、NUMERIC、INT、FLOAT、DOUBLE、VARCHAR、TIMESTAMP。
  - CAST() and CONVERT (... USING...) are standard SQL syntax.
  - CAST (str AS BINARY) is equivalent to BINARY str.
  - CAST (expr AS CHAR) treats expressions as strings in the default character set.
  - In CAST (expr AS float (M, D)) and CAST (expr AS double (M, D)), the maximum value of M is 255 and the maximum value of D is 30.
  - CAST (expr AS Float (X)) specifies the length, and when X<24, it is processed according to the float; When 24<X<=53, handle according to the maximum length and accuracy of double

be careful

Using the CAST() function to change the column type to DATE, DATETIME, or TIME simply identifies the column to a specified data type, rather than changing its value.

The final execution result of CAST() will be converted to the specified column type.

When querying, using cast to convert data to varchar (0) will output an empty string, and using create table as select from to query non empty columns from existing tables to convert varchar to create a new table. If non empty columns are converted to varchar (0), an error will be reported.

Given that the timestamp type will support microsecond precision in the future, the current processing method for cast as timestamp is to support converting '2020 01 02 11:11:11:12123451' to timestamp. However, due to the storage part still not supporting it, create as select cast (... as timestamp) only supports truncation to the second level.

The function can convert the UTC time format string of '1970-01-01 00:00:01~2038-01-10 03:14:07' to a timestamp type, but the maximum timestamp storage value is, 2038-01-01 00:59:59.

Example

Example 1: Convert NOW () to DATE type.

```
gbase> SELECT CAST(NOW() AS DATE) FROM dual;
+---------------------+
| CAST(NOW() AS DATE) |
+---------------------+
| 2020-04-01          |
+---------------------+
1 row in set
```

Example 2: The conversion of string and number types is an implicit operation, and users only need to treat the string value as a number when using it.

```
gbase> SELECT 1+'1' FROM dual;
+-------+
| 1+'1' |
+-------+
|     2 |
+-------+
1 row in set
```

Example 3: CAST (str AS BINARY) is equivalent to BINARY str.

```
gbase> SELECT CAST('a' AS BINARY) = 'a ' FROM dual;
+----------------------------+
| CAST('a' AS BINARY) = 'a ' |
+----------------------------+
|                          0 |
+----------------------------+
1 row in set
gbase>    SELECT 'A' = 'a ';
+------------+
| 'A' = 'a ' |
+------------+
|          1 |
+------------+
1 row in set (Elapsed: 00:00:00.00)
gbase> SELECT BINARY 'A' = 'a ' FROM dual;
+-------------------+
| BINARY 'A' = 'a ' |
+-------------------+
|                 0 |
```

```
+-------------------+

1 row in set
```

Example 4: CAST (str AS varchar (X)) Example

```
gbase> select cast('1.2345' as varchar) as data_ varchar;
    +--------------+
    | data_ varchar |
    +--------------+
    | 1.2345            |
    +--------------+
    1 row in set (Elapsed: 00:00:00.00)

gbase> select cast('1.2345' as varchar(10)) as data_ varchar;
    +--------------+
    | data_ varchar |
    +--------------+
    | 1.2345            |
    +--------------+
    1 row in set (Elapsed: 00:00:00.00)

gbase> select cast('1.2345' as varchar(3)) as data_ varchar;
    +--------------+
    | data_ varchar |
    +--------------+
    | 1.2                |
    +--------------+
    1 row in set, 1 warning (Elapsed: 00:00:00.00)

gbase> select cast('1.2345' as varchar(0)) as data_ varchar;
    +--------------+
    | data_ varchar |
    +--------------+
    |                      |
    +--------------+
    1 row in set, 1 warning (Elapsed: 00:00:00.01)

gbase> create table t3 as select cast(a as varchar) as a from t;
    Query OK, 2 rows affected (Elapsed: 00:00:00.51)

gbase> desc t3;
    +-------+-------------+------+-----+---------+-------+
    | Field | Type              | Null | Key | Default | Extra |
    +-------+-------------+------+-----+---------+-------+
```

```
| a | varchar(11) | YES | | NULL | | |
+-------+-------------+------+-----+---------+-------+
1 row in set (Elapsed: 00:00:00.00)

gbase> create table t4 as select cast(a as varchar(0)) as a from t;
ERROR 1705 (HY000): gcluster DML error:
[192.168.146.21:5050](GBA-02AD-0005)Failed to query in gnode:
DETAIL: Truncated incorrect CHAR(0) value: '1'
SQL: SELECT /*192.168.146.20_ 6_ 31_ 2021-01-14_ 15:25:42*/ /*+
TID('111') */ cast(`vcname000001.testdb.t`.`a` as char(0)) AS `a` FROM
`testdb`.` t_ n1` `vcname000001.testdb.t` target into server (HOST
'192.168.146.21,192.168.146.20', PORT 5050, USER 'root', PASSWORD '',
DATABASE 'testdb', TABLE 't4_n1', COMMENT 'col_seq 0, table_host 0 0
1, scn 18, distribution 1' )
```

5.1.5.1.5.3 TO_ SINGLE_ BYTE

grammar

```
TO_ SINGLE_ BYTE(arg)
```

Function Description

Convert the incoming arg from a full width character to a half width character. Arg can be any type of value and column. If arg is a string and the string contains full width, the full width character will be converted to half width character in the output result, and other characters will remain unchanged.

This function is only valid for UTF8 and GBK character sets.

- Currently, only 95 characters support full width to half width conversion.

  The 95 characters are as follows:

Space	!	"	#	$	%	&	‘	(	)
*	+	,	−	.	/	:	;	<	=
>	?	@	[	\	]	^	_	‘	{
|	}	~	A-Z	a-z	0-9				

  When creating as select, the field type containing the function column is determined based on the field type of the query result. If the field type of the query result is a character type, the varchar or longblob types will be determined based on the maximum length of the result.

be careful

- Only VARCHAR, CHAR, and TEXT support string type column types that support full width characters, and use to_ single_ Byte conversion succeeded.
- Although LONGBLOB and BLOB can store full width characters, they are stored in binary format_ SINGLE_ After BYTE conversion, it is still a full width character.
- BLOB type approved by TO_ SINGLE_ Convert BYTE to VARBINARY type

Example

Example:

```
create table t(a int, b varchar(10), c datetime, t text, e longblob, f blob, g char(10));
gbase> insert into t values(1, 'aaaaa a ', '2011-01-01 11:11:11', ' a a a a ', ' a a a a ',
' a a a a ', ' a a a a ');
Query OK, 1 row affected (Elapsed: 00:00:00.05)

gbase> select to_ single_ byte(a) as sing_ a,to_ single_ byte(b) as sing_ b, to_ single_
byte(c) as sing_ c, to_ single_ byte(t) as sing_ t, to_ single_ byte(e) as sing_ e, to_
single_ byte(f) as sing_ f, to_ single_ byte(g) as sing_ g from t;
+--------+--------+---------------------+--------+--------------+--------------+------------+
| sing_ a | sing_ b | sing_ c                      | sing_ t | sing_ e            | sing_ f           |
sing_ g         |
+--------+--------+---------------------+--------+--------------+--------------+------------+
| 1          | aaaaaa | 2011-01-01 11:11:11 | aaaa     | a a a a        | a a a a        |
aaaa            |
+--------+--------+---------------------+--------+--------------+--------------+------------+
1 row in set (Elapsed: 00:00:00.03)

gbase> create table ty as select to_ single_ byte(a) as sing_ a,to_ single_ byte(b) as
sing_ b, to_ single_ byte(c) as sing_ c, to_ single_ byte(t) as sing_ t, to_ single_ byte(e)
as sing_ e, to_ single_ byte(f) as sing_ f, to_ single_ byte(g) as sing_ g from t;
Query OK, 1 row affected (Elapsed: 00:00:00.11)

gbase>    show create table ty \G
*************************** 1. row ***************************
           Table: ty
Create Table: CREATE TABLE "ty" (
  "sing_a" varchar(11) DEFAULT NULL,
```

```
  "sing_b" varchar(10) DEFAULT NULL,
  "sing_c" varchar(26) DEFAULT NULL,
  "sing_t" varchar(10922) DEFAULT NULL,
  "sing_e" longblob,
  "sing_f" varbinary(32767) DEFAULT NULL,
  "sing_g" varchar(10) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_ tablespace'
1 row in set (Elapsed: 00:00:00.00)

gbase> select * from ty;
+--------+--------+---------------------+--------+--------------+--------------+------------+
| sing_ a | sing_ b | sing_ c                    | sing_ t | sing_ e            | sing_ f           |
sing_ g         |
+--------+--------+---------------------+--------+--------------+--------------+------------+
| 1          | aaaaaa | 2011-01-01 11:11:11 | aaaa     | a a a a        | a a a a        |
aaaa            |
+--------+--------+---------------------+--------+--------------+--------------+------------+
1 row in set (Elapsed: 00:00:00.02)
```

5.1.5.1.6 Date arithmetic operation

Grammar Description

```
Date+(-) bit_ expr
```

Equivalent to the following syntax:

```
Date+(-) interval expr type
```

Operation Description

The logic of date addition and subtraction is the same as that of ordinary addition and subtraction operations, except that the number, character, or expression added after it is in days. This syntax adds (or subtracts) the specified bit after variables of type date, date time, or timestamp_ Expr's number of days.

Example

Example 1: CAST ('2019-06-18 'as date)+30 is the date, and returns the date after adding 30 days.

```
gbase> SELECT CAST('2019-06-18' as date)    + 30    FROM dual;
+----------------------------------+
| CAST('2019-06-18' as date)    + 30 |
+----------------------------------+
```

```
| 2019-07-18                        |
+-----------------------------------+
1 row in set
```

5.1.5.2 Control flow function

5.1.5.2.1 CASE

grammar

```
CASE value WHEN [compare-value] THEN result [WHEN [compare-value] THEN result ...] [ELSE result] END
```

Function Description

Match one by one, and when value=compare value is met, return the corresponding result. If no match is found, return the result after ELSE. If there is no ELSE clause, it returns NULL by default.

If there is overlap between compare values in the condition, that is, when the value value satisfies multiple compare value conditions, only the first value that satisfies is returned.

grammar

```
CASE WHEN [condition] THEN result [WHEN [condition] THEN result ...] [ELSE result] END
```

Function Description

One by one, when the condition is TRUE, the corresponding result is returned. If all conditions are FALSE, the result after ELSE is returned. If there is no ELSE clause, it returns NULL by default.

The default return value type of a CASE expression is the compatible set type of all return values, depending on its context:

If used in a string context, the returned result is a string;

If used in a numerical context, returns a real or integer value with a decimal value as the result.

Example

Example 1: value=compare value, returns the corresponding result value.

```
gbase> SELECT CASE 1 WHEN 1 THEN 'one' WHEN 2 THEN 'two' ELSE
'more' END FROM dual;
+-------------------------------------------------------------+
| CASE 1 WHEN 1 THEN 'one' WHEN 2 THEN 'two' ELSE 'more' END  |
+-------------------------------------------------------------+
| one                                                         |
+-------------------------------------------------------------+
1 row in set
```

Example 2: When the condition is TRUE, the corresponding result value is returned.

```
gbase> SELECT CASE WHEN 1>0 THEN 'true' ELSE 'false' END FROM
dual;
+-----------------------------------------+
| CASE WHEN 1>0 THEN 'true' ELSE 'false' END  |
+-----------------------------------------+
| true                                    |
+-----------------------------------------+
1 row in set
```

Example 3: Value is not equal to compare value, and the return value is NULL.

```
gbase> SELECT CASE 'c' WHEN 'a' THEN 1 WHEN 'b' THEN 2 END
FROM dual;
+-----------------------------------------+
| CASE 'c' WHEN 'a' THEN 1 WHEN 'b' THEN 2 END  |
+-----------------------------------------+
|                                    NULL |
+-----------------------------------------+
1 row in set
```

5.1.5.2.2 DECODE

grammar

```
DECODE(value,value1,result1, value2,result2, value3,result3,... , result)
```

Function Description

Similar to CASE value when value1 THEN result1, The only difference is that if value is a NULL value, it can match the subsequent NULL values.

Example

Example 1: If there is no matching value, the return value is result.

```
gbase> SELECT DECODE(5,1,10,2,20,3,30,4,40, 50) FROM dual;
+-------------------------------------+
| DECODE(5,1,10,2,20,3,30,4,40, 50) |
+-------------------------------------+
|                                  50 |
+-------------------------------------+
1 row in set
```

Example 2: Value is an expression that matches value1 and returns the value result1.

```
gbase> SELECT DECODE( (2 * 5) ,10,100,20,200,600) FROM dual;
+--------------------------------------+
| DECODE( (2 * 5) ,10,100,20,200,600) |
+--------------------------------------+
|                                  100 |
+--------------------------------------+
1 row in set
```

5.1.5.2.3 IF(expr1,expr2,expr3)

grammar

```
IF(expr1,expr2,expr3)
```

Function Description

If expr1 is true, the return value of IF() is expr2. If expr1 takes the value of FALSE, 0, or NULL, the return value is expr3.

The return value rules of IF() are the same as those of CASE expressions.

Example

Example 1: The value of expre1 is FALSE, and the return value is expr3.

```
gbase> SELECT IF(FALSE,2,3) FROM dual;
+-------------+
| IF(FALSE,2,3) |
+-------------+
|             3 |
+-------------+
1 row in set
```

Example 2: The value of expre1 is TRUE, and the return value is expr2.

```
gbase> SELECT IF(TRUE,'yes','no') FROM dual;
+--------------------+
| IF(TRUE,'yes','no') |
+--------------------+
| yes                |
+--------------------+
1 row in set
```

Example 3: expr1 is an expression with a value of TRUE and a return value of expr2.

```
gbase> SELECT IF(1<2,'no','yes') FROM dual;
+---------------------------------------+
| IF(1<2,'no','yes')                    |
+---------------------------------------+
| no                                    |
+---------------------------------------+
1 row in set
```

Example 4: expr1 is an expression, the value is not true, and the return value is expr3.

```
gbase> SELECT IF(1>2,NULL,'no') FROM dual;
+-------------------+
| IF(1>2,NULL,'no') |
+-------------------+
| no                |
+-------------------+
1 row in set
```

5.1.5.2.4 IFNULL(expr1,expr2)

grammar

```
IFNULL(expr1,expr2)
```

Function Description

If expr1 is not NULL, the return value of IFNULL() is expr1, otherwise its return value is expr2.

The return value of IFNULL() is either a number or a string, depending on the context in which it is used. Equivalent to IF (expre1, expre1, expre2).

Example

Example 1: expr1 is not NULL, and the return value is expr1.

```
gbase> SELECT IFNULL(1,0) FROM dual;
+-------------+
| IFNULL(1,0) |
+-------------+
|           1 |
+-------------+
1 row in set
```

Example 2: expr1 is NULL and the return value is expr2.

```
gbase> SELECT IFNULL(NULL,10) FROM dual;
+-----------------+
| IFNULL(NULL,10) |
+-----------------+
|              10 |
+-----------------+
1 row in set
```

5.1.5.2.5 NULLIF(expr1,expr2)

grammar

```
NULLIF(expr1,expr2)
```

Function Description

If expr1=expr2 holds, the return value is NULL, otherwise the return value is expr1.

Equivalent to CASE WHERE expr1=expr2 THEN NULL ELSE expr1 END.

Example

Example 1: expr1=expr2, with a return value of NULL.

```
gbase> SELECT NULLIF(1,1) FROM dual;
+-------------+
| NULLIF(1,1) |
+-------------+
|        NULL |
+-------------+
1 row in set
```

Example 2: expr1= Expr2, the return value is expr1.

```
gbase> SELECT NULLIF(1,2) FROM dual;
+-------------+
| NULLIF(1,2) |
+-------------+
|           1 |
+-------------+
1 row in set
```

5.1.5.3 String function

summary

For functions that operate on string positions, the first position is marked as 1.

5.1.5.3.1 ASCII(str)

Function Description

- Returns the ASCII value of the first character of the string str;
- If str is an empty string, the return value is 0;
- If str is a NULL, the return value is NULL;
- ASCII () is only suitable for characters with values between 0 and 255.

Example

Example 1: str has a value of "2" and returns the ASCII code value corresponding to "2".

```
gbase> SELECT ASCII('2') FROM dual;
+------------+
| ASCII('2') |
+------------+
|         50 |
+------------+
1 row in set
```

Example 2: str has a value of "dx" and returns the ASCII code value corresponding to "d".

```
gbase> SELECT ASCII('dx') FROM dual;
+-------------+
| ASCII('dx') |
+-------------+
|         100 |
```

```
+-------------+
1 row in set
```

5.1.5.3.2 BIN(N)

Function Description

- Returns the binary form of N, where N is a BIGINT type numerical value;
- If N is a NULL, the return value is NULL.

Example

Example 1: N has a value of "12" and returns the binary form corresponding to "12".

```
gbase> SELECT BIN(12) FROM dual;
+---------+
| BIN(12) |
+---------+
| 1100    |
+---------+
1 row in set
```

5.1.5.3.3 BIT_ LENGTH(str)

Function Description

Returns the bit length of the string str, calculated in bits.

Example

Example 1: The value of str is "text", which returns its corresponding bit length.

```
gbase> SELECT BIT_ LENGTH('text') FROM dual;
+--------------------+
| BIT_ LENGTH('text') |
+--------------------+
|                 32 |
+--------------------+
1 row in set
```

Example 2: If the current character set is UTF8 and str is "Nanjing University Universal", return its corresponding bit length.

```
gbase> SELECT BIT_ LENGTH ('Nanjing University General Motors') from
dual;
```

```
+----------------------------+
| BIT_ LENGTH|
+----------------------------+
|                         96 |
+----------------------------+
1 row in set

gbase> SHOW VARIABLES LIKE 'CHARACTER_ SET_ SERVER';
+----------------------+-------+
| Variable_ name               | Value |
+----------------------+-------+
| character_ set_ server | utf8    |
+----------------------+-------+
1 row in set
```

5.1.5.3.4 CHAR(N1,N2...)

Function Description

N1, N2... are integer type parameters that return a string composed of characters corresponding to the ASCII code value represented. If the parameter list contains NULL, it is ignored.

Example

Example 1: The value of N is "71,66,97115101", and the ASCII codes corresponding to each integer represent characters such as "G", "B", "a", "s", and "e".

```
gbase> SELECT CHAR(71,66,97,115,101) FROM dual;
+-----------------------+
| CHAR(71,66,97,115,101) |
+-----------------------+
| GBase                          |
+-----------------------+
1 row in set
```

Example 2: If the value of N contains NULL, then NULL is ignored.

```
gbase> SELECT CHAR(77,72,NULL,'77') FROM dual;
+----------------------+
| CHAR(77,72,NULL,'77') |
+----------------------+
| MHM                            |
+----------------------+
```

```
1 row in set
```

5.1.5.3.5 CHAR_ LENGTH(str)

Function Description

Returns the character length of the string str, in characters.

Example

Example 1: Returns the character length of "text".

```
gbase> SELECT CHAR_ LENGTH('text') FROM dual;
+---------------------+
| CHAR_ LENGTH('text') |
+---------------------+
|                    4 |
+---------------------+
1 row in set
```

Example 2: Returns the character length of "Nanjing University General".

```
gbase> SELECT CHAR_ LENGTH ('Nanjing University General Motors')
from dual;
+--------------------------+
| CHAR_ LENGTH|
+--------------------------+
|                        4 |
+--------------------------+
1 row in set
```

5.1.5.3.6 CHARACTER_ LENGTH(str)

Function Description

Equivalent to CHAR_ LENGTH()。

5.1.5.3.7 CONCAT(str1,str2,...)

Function Description

The return result is a string generated by the connection parameter. If any parameter is NULL, the return value is NULL.

Example

Example 1: Connection strings "GB", "a", "se".

```
gbase> SELECT CONCAT('GB', 'a', 'se') FROM dual;
+-------------------------+
| CONCAT('GB', 'a', 'se') |
+-------------------------+
| GBase                   |
+-------------------------+
1 row in set
```

Example 2: If any parameter is NULL, the return value is NULL.

```
gbase> SELECT CONCAT('GB', NULL, 'se') FROM dual;
+-------------------------+
| CONCAT('GB', NULL, 'se') |
+-------------------------+
| NULL                    |
+-------------------------+
1 row in set
```

Example 3: If a number is used in a string context, it will be automatically converted to a string.

```
gbase> SELECT CONCAT('hello you ',2) FROM dual;
+------------------------+
| CONCAT('hello you ',2) |
+------------------------+
| hello you 2            |
+------------------------+
1 row in set
```

Example 4: 'str1 | | str2 | | str3' is equivalent to CONCAT (str1, str2, str3).

```
gbase> SELECT 'GB' || 'a' || 'se' FROM dual;
+--------------------+
| 'GB' || 'a' || 'se' |
+--------------------+
| GBase              |
+--------------------+
1 row in set
```

5.1.5.3.8 CONCAT_ WS(separator,str1,str2,...)

Function Description

CONCAT_ WS() stands for CONCAT With separator, which is a special form of CONCAT(). The first parameter is the separator for other parameters, which can be a character, a string, or a parameter. If the delimiter is NULL, the result is NULL, and the function ignores the NULL value in the parameter after the delimiter.

Example

Example 1: The separator is', '.

```
gbase> SELECT CONCAT_ WS(',','First name','Second name','Last Name')
FROM dual;
+-----------------------------------------------------+
| CONCAT_ WS(',','First name','Second name','Last Name') |
+-----------------------------------------------------+
| First name,Second name,Last Name                    |
+-----------------------------------------------------+
1 row in set
```

Example 2: The separator is', 'and one of the strs is NULL.

```
gbase> SELECT CONCAT_ WS(',','First name',NULL,'Last Name') FROM
dual;
+-----------------------------------------------+
| CONCAT_ WS(',','First name',NULL,'Last Name') |
+-----------------------------------------------+
| First name,Last Name                          |
+-----------------------------------------------+
1 row in set
```

5.1.5.3.9 CONV(N,from_base,to_base)

Function Description

Conversion between different decimal systems. Convert N from_ Convert base to to to_ Base, return value is to_ A string in base format. If any parameter is NULL, the return value is NULL. The parameter N is an integer or string, with a minimum of 2 decimal and a maximum of 36 decimal. If to_ If base is a negative number, N is considered a signed number, otherwise N is considered an unsigned number.

CONV (N, 10,2) is equivalent to BIN (N).

Example

Example 1: Convert 'a' from hexadecimal to binary.

```
gbase> SELECT CONV('a',16,2) FROM dual;
+----------------+
| CONV('a',16,2) |
+----------------+
| 1010           |
+----------------+
1 row in set
```

Example 2: Convert "6E" from base 18 to base 8.

```
gbase> SELECT CONV('6E',18,8) FROM dual;
+-----------------+
| CONV('6E',18,8) |
+-----------------+
| 172             |
+-----------------+
1 row in set
```

Example 3: Convert "-17" from decimal to -18.

```
gbase> SELECT CONV(-17,10,-18) FROM dual;
+------------------+
| CONV(-17,10,-18) |
+------------------+
| -H               |
+------------------+
1 row in set
```

Example 4: Convert "10+'10 '+'10'+0xa" from decimal to decimal.

```
gbase> SELECT CONV(10+'10'+'10'+0xa,10,10) FROM dual;
+-----------------------------+
| CONV(10+'10'+'10'+0xa,10,10) |
+-----------------------------+
| 40                          |
+-----------------------------+
1 row in set
```

5.1.5.3.10 ELT(N,str1,str2,str3,...)

Function Description

Returns the Nth str. If N=1, the return value is str1, if N=2, the return value is str2, and so on. If N is less than 1 or greater than the number of parameters, the return value is NULL.

Example

Example 1: If N=1, the return value is str1.

```
gbase> SELECT ELT(1, 'ej', 'Heja', 'hej', 'foo') FROM dual;
+-----------------------------------+
| ELT(1, 'ej', 'Heja', 'hej', 'foo') |
+-----------------------------------+
| ej                                |
+-----------------------------------+
1 row in set
```

Example 2: If N=4, the return value is str4.

```
gbase> SELECT ELT(4, 'ej', 'Heja', 'hej', 'foo') FROM dual;
+-----------------------------------+
| ELT(4, 'ej', 'Heja', 'hej', 'foo') |
+-----------------------------------+
| foo                               |
+-----------------------------------+
1 row in set
```

Example 3: N=9, greater than the number of parameters, returns a value of NULL.

```
gbase> SELECT ELT(9, 'ej', 'Heja', 'hej', 'foo') FROM dual;
+-----------------------------------+
| ELT(9, 'ej', 'Heja', 'hej', 'foo') |
+-----------------------------------+
| NULL                              |
+-----------------------------------+
1 row in set
```

5.1.5.3.11 EXPORT_ SET

Grammar format

```
EXPORT_ SET(bits,on,off[,separator[,number_of_bits]])
```

Function Description

The return value is a string, and the bits of the parameter bits are detected in order from right to left (from low bit to high bit). For each bit value in the value, if it is 1, an on string is obtained, and if it is 0, an off string is obtained. The string is separated by the parameter separator (default is comma ","), number_ of_ Bits represents the number of binary digits being checked (default is 64).

Example

Example 1: The value of parameter bits is "5", and the corresponding binary is 0101. When detected from right to left, the output is 1010, and the corresponding ON and OFF values are "Y" and "N". Therefore, the output is "Y, N, Y, N".

```
gbase> SELECT EXPORT_ SET(5,'Y','N',',',4) FROM dual;
+-----------------------------+
| EXPORT_ SET(5,'Y','N',',',4) |
+-----------------------------+
| Y,N,Y,N                     |
+-----------------------------+
1 row in set
```

Example 2: number_ of_ When the number of bits is greater than the binary digits corresponding to the bits value, the off value is used to fill in, that is, "0".

```
gbase> SELECT EXPORT_ SET(6,'1','0',',',10) FROM dual;
+-------------------------------+
| EXPORT_ SET(6,'1','0',',',10) |
+-------------------------------+
| 0,1,1,0,0,0,0,0,0,0           |
+-------------------------------+
1 row in set
```

5.1.5.3.12 FIELD(str,str1,str2,str3,...)

Function Description

Returns the position of a string equal to str. If str is equal to str1, return 1; if str is equal to str2, return 2, and compare backwards in sequence. When all are unequal, the return value is 0; If all parameters for FIELD() are strings, then all parameters are compared according to the string; If all parameters are numerical values, compare them according to the numerical values; If str is NULL, the return value is 0 because NULL cannot be compared equally with any value. FIELD() is the complement of ELT().

Example

Example 1: The parameter of FIELD() is a string, and all parameters are compared based on the string.

```
gbase> SELECT FIELD('ej','Hej', 'ej', 'Heja', 'hej', 'foo') FROM dual;
+-----------------------------------------------+
| FIELD('ej','Hej', 'ej', 'Heja', 'hej', 'foo') |
+-----------------------------------------------+
|                                             2 |
+-----------------------------------------------+
```

```
1 row in set
```

Example 2: The parameters of FIELD() are numbers, and all parameters are compared based on numbers.

```
gbase> SELECT FIELD('112','12','112','123','213') FROM dual;
+--------------------------------------+
| FIELD('112','12','112','123','213') |
+--------------------------------------+
|                                    2 |
+--------------------------------------+
1 row in set
```

Example 3: str and str1 The strns are not equal, and the return value is 0.

```
gbase> SELECT FIELD('fo','Hej', 'ej', 'Heja', 'hej', 'foo') FROM dual;
+-------------------------------------------------+
| FIELD('fo','Hej', 'ej', 'Heja', 'hej', 'foo') |
+-------------------------------------------------+
|                                               0 |
+-------------------------------------------------+
1 row in set
```

5.1.5.3.13 FIND_ IN_ SET(str,strlist)

Function Description

Returns the position of the string str in the strlist. The parameter strlist consists of multiple substrings separated by the character ','; If the string str is in the strlist, return the matching position, starting from 1; If the string str is not in strlist or strlist is an empty string, the return value is 0; If str is NULL, the return value is NULL and cannot be compared equally with any value.

Example

Example 1: The string str in strlist returns its corresponding position.

```
gbase> SELECT FIND_ IN_ SET('b','a,b,c,d') FROM dual;
+-----------------------------+
| FIND_ IN_ SET('b','a,b,c,d') |
+-----------------------------+
|                           2 |
+-----------------------------+
1 row in set
```

5.1.5.3.14 HEX(N_or_S)

Function Description

Returns the hexadecimal value corresponding to the parameter. If N_ or_ If S is a number, it returns its hexadecimal string form. Here, N is a BIGINT number, equivalent to CONV (N, 10,16); If N_ or_ If S is a string, it returns the hexadecimal form corresponding to each character, where each character is converted to two hexadecimal digits. The string appearing in the form of 0xff is the inversion operation of this function, where every two hexadecimal digits are converted into their ASCII code and output one character.

Example

Example 1: N_ or_ The S value is a number.

```
gbase> SELECT HEX(255) FROM dual;
+----------+
| HEX(255) |
+----------+
| FF       |
+----------+
1 row in set
```

Example 2: N_ or_ The S value is a string.

```
gbase> SELECT HEX('abc') FROM dual;
+------------+
| HEX('abc') |
+------------+
| 616263     |
+------------+
1 row in set
```

Example 3: A string in the form of 0xff is an inversion operation of the HEX (N_or_S) function.

```
gbase> SELECT 0x616263 FROM dual;
+----------+
| 0x616263 |
+----------+
| abc      |
+----------+
1 row in set
```

5.1.5.3.15 INSERT(str,pos,len,newstr)

Function Description

In the string str, starting from the pos position, select a substring with a length of len characters and replace it with the string newstr. If the pos value is not within the length range, the original string is returned; If the len value is not within the remaining length range of the string, replace the remaining strings starting from the pos position; If any parameter is NULL, it returns NULL.

Example

Example 1: Replace the four characters "adra" starting from the third position of the string "Quadratic" with "What".

```
gbase> SELECT INSERT('Quadratic', 3, 4, 'What') FROM dual;
+----------------------------------+
| INSERT('Quadratic', 3, 4, 'What') |
+----------------------------------+
| QuWhattic                        |
+----------------------------------+
1 row in set
```

Example 2: If the value of pos is not within the length range, the original string is returned.

```
gbase> SELECT INSERT('Quadratic', -1, 4, 'What') FROM dual;
+-----------------------------------+
| INSERT('Quadratic', -1, 4, 'What') |
+-----------------------------------+
| Quadratic                         |
+-----------------------------------+
1 row in set
```

Example 3: If len is not within the remaining length range of the string, replace the remaining strings starting from position 3 with 'What'.

```
gbase> SELECT INSERT('Quadratic', 3, 100, 'What') FROM dual;
+------------------------------------+
| INSERT('Quadratic', 3, 100, 'What') |
+------------------------------------+
| QuWhat                             |
+------------------------------------+
1 row in set
```

5.1.5.3.16 INSTR()

Grammar format

```
INSTR(str,substr)
INSTR(str,substr,start_position,N)
```

Table 510 Parameter Description

parameter	explain
str	String mother string.
substr	String substring.
start_ position	Represents matching starting from the first character of the string (left end). If it is a negative number, the substring is searched backwards from the right end. Optional parameter, default to 1.
nth_ appearance	From start_ Position starts searching for the matching string at the end of the string. Optional parameter, default to 1.

Function Description

- The first grammatical function:

Returns the position where the substr substr first appears in the string str (starting from the left end). If not, return 0. If one parameter is NULL, it returns NULL.

- The second grammatical function:

1. Find nth_ The function of appearance matching strings;
2. From the start of the parent string_ The function of starting with a position character to search for a matching string;
3. Support for the third parameter start_ When the position is negative, that is, starting from the | start on the right side of the parent string_ Position | The function of reverse searching substrings for positions.

> **explain**
>
> The position of substr in str, counted starting with 1.
>
> The returned position is counted in the forward direction of the entire string, regardless of which position it starts from

Example

Example 1: Returns the position where "bar" first appears in "foobarbar".

```
gbase> SELECT INSTR('foobarbar', 'bar') FROM dual;
+--------------------------+
| INSTR('foobarbar', 'bar') |
+--------------------------+
|                        4 |
+--------------------------+
1 row in set
```

Example 2: 'foobar' is not in 'xbar'.

```
gbase> SELECT INSTR('xbar', 'foobar') FROM dual;
+------------------------+
| INSTR('xbar', 'foobar') |
+------------------------+
|                      0 |
+------------------------+
1 row in set
```

Example 3: If any parameter is a binary string, it is case sensitive.

```
gbase> SELECT INSTR('foobarbar', BINARY 'Bar') FROM dual;
+---------------------------------+
| INSTR('foobarbar', BINARY 'Bar') |
+---------------------------------+
|                               0 |
+---------------------------------+
1 row in set
```

Example 4: Starting from the third character of the string "beiwelcome to beijing", search for the position where "bei" first appears.

```
gbase> SELECT INSTR ('beiwelcometobeijing','bei',3) FROM dual;
+-----------------------------------+
| INSTR('beiwelcometobeijing','bei',3) |
+-----------------------------------+
|                                13 |
+-----------------------------------+
1 row in set
```

Example 5: Start matching from the first character of the string "1121112222233333" and find the position where the second "11" appears.

```
gbase> SELECT INSTR ('112111222222333333','11',1,2) FROM dual;
+--------------------------------------+
| INSTR('112111222222333333','11',1,2) |
+--------------------------------------+
```

```
|                                                4 |
+-----------------------------------------+
1 row in set
```

Example 6: Start matching from the penultimate character of the string "welcome to china" and find the position where the first "e" appears.

```
gbase> SELECT INSTR ('welcometochina','e',-1,1) FROM dual;
+----------------------------------+
| INSTR('welcometochina','e',-1,1) |
+----------------------------------+
|                                7 |
+----------------------------------+
1 row in set
```

5.1.5.3.17 LCASE(str)

Function Description

Change all characters in the string str to lowercase.

Example

Example 1: Convert 'QUADRATICALLY' to lowercase.

```
gbase> SELECT LCASE('QUADRATICALLY') FROM dual;
+-------------------------+
| LCASE('QUADRATICALLY') |
+-------------------------+
| quadratically           |
+-------------------------+
1 row in set
```

5.1.5.3.18 LEFT(str,len)

Function Description

Returns the leftmost len character in the string str. If len exceeds the length of str or is less than 1, it returns null; If len is NULL, it is returned as NULL.

Example

Example 1: Returns the left five characters of 'foobar'.

```
gbase> SELECT LEFT('foobarbar', 5) FROM dual;
```

```
+----------------------+
| LEFT('foobarbar', 5) |
+----------------------+
| fooba                |
+----------------------+
1 row in set
```

5.1.5.3.19 LENGTH(str)

Function Description

Returns the length of the string str, calculated in bytes.

Example

Example 1: Returns the byte length of "text".

```
gbase> SELECT LENGTH('text') FROM dual;
+----------------+
| LENGTH('text') |
+----------------+
|              4 |
+----------------+
1 row in set
```

Example 2: Returns the byte length (utf8 encoding) of "Nanjing University General".

```
Gbase>SELECT LENGTH ('Nanjing University General Motors') from dual;
+------------------------+
|LENGTH|
+------------------------+
|                     12 |
+------------------------+
1 row in set
```

5.1.5.3.20 LOAD_ FILE(file_name)

Function Description

Read the file and return it in string format. If the file does not exist or cannot be read, the function returns a value of NULL.

Example

Example 1: Read the a.txt file and return the content.

```
gbase> SELECT LOAD_ FILE('/home/gbase/a.txt') FROM dual;
+---------------------------------+
| LOAD_ FILE('/home/gbase/a.txt') |
+---------------------------------+
| adfdfafgagsdgewr                |
+---------------------------------+
1 row in set
```

5.1.5.3.21 LOCATE()

Grammar format

```
LOCATE(substr,str)
LOCATE(substr,str,pos)
```

Function Description

- The first syntax returns the position where the substr substr first appears in the string str. LOCATE (substr, str) is similar to INSTR (str, substr), except that the position of the parameter is reversed;
- The second syntax returns the position where the substr substr first appears after the pos position in the string str.

> **explain**
>
> - If substr is not in str, return 0;
> - The position of substr in str, counting from 1;
> - If the pos value is greater than the length of str, it returns 0;
> - If pos is NULL, returns NULL.

Example

Example 1: Returns the position where "bar" first appears in "foobarbar".

```
gbase> SELECT LOCATE('bar', 'foobarbar') FROM dual;
+----------------------------+
| LOCATE('bar', 'foobarbar') |
+----------------------------+
|                          4 |
+----------------------------+
1 row in set
```

Example 2: 'xbar' is not in 'foobar' and the return value is 0.

```
gbase> SELECT LOCATE('xbar', 'foobar') FROM dual;
+--------------------------+
| LOCATE('xbar', 'foobar') |
```

```
+--------------------------+
|                        0 |
+--------------------------+
1 row in set
```

Example 3: Returns the position where "bar" first appears after the 5th position in "foobarbar".

```
gbase> SELECT LOCATE('bar', 'foobarbar',5) FROM dual;
+------------------------------+
| LOCATE('bar', 'foobarbar',5) |
+------------------------------+
|                            7 |
+------------------------------+
1 row in set
```

Example 4: If any parameter is a binary string, it is case sensitive.

```
gbase> SELECT LOCATE(BINARY'bAr', 'foobarbar',5) FROM dual;
+-----------------------------------+
| LOCATE(BINARY'bAr', 'foobarbar',5) |
+-----------------------------------+
|                                 0 |
+-----------------------------------+
1 row in set
```

5.1.5.3.22 LOWER(str)

Function Description

Change all characters in the string str to lowercase.

Example

Example 1: POWER (str) is equivalent to LCASE ().

```
gbase> SELECT
LOWER('QUADRATICALLY'),LCASE('QUADRATICALLY') FROM
dual;
+------------------------+------------------------+
| LOWER('QUADRATICALLY') | LCASE('QUADRATICALLY') |
+------------------------+------------------------+
| quadratically          | quadratically          |
+------------------------+------------------------+
1 row in set
```

5.1.5.3.23 LPAD(str,len,padstr)

Function Description

Fill the left side of str with the string padstr until its length reaches len characters, and then return the filled str; If the length of str is longer than len, it will be truncated to len characters.

Example

Example 1: Add "??" to the left of "hi", with a total length of 4 digits.

```
gbase> SELECT LPAD ('hi',4,'??') FROM dual;
+-------------------+
| LPAD('hi',4,'??') |
+-------------------+
| ?? hi             |
+-------------------+
1 row in set
```

Example 2: If the length of 'hi' is greater than 1, 'hi' will be truncated to 1 character.

```
gbase> SELECT LPAD('hi',1,'??') FROM dual;
+-------------------+
| LPAD('hi',1,'??') |
+-------------------+
| h                 |
+-------------------+
1 row in set
```

5.1.5.3.24 LTRIM(str)

Function Description

Remove multiple consecutive spaces to the left of str.

Example

Example 1: Remove the two spaces to the left of 'bar'.

```
gbase> SELECT LTRIM('  barbar') FROM dual;
+------------------+
| LTRIM('  barbar') |
+------------------+
| barbar           |
```

```
+-------------------+
1 row in set
```

5.1.5.3.25 MAKE_ SET(bits,str1,str2,...)

Function Description

Returns a set value (a string composed of strs separated by ","), consisting of a string with corresponding bits in the bits set as 1. The bit values in the bits are tested in order from right to left (from low bit to high bit). If the first bit value corresponding to str1 is 1 and the second bit value corresponding to str2 is 1, then str1 and str2 are returned. And so on, str1, str2 The NULL value in will not be added to the result.

Example

Example 1: Verify the bit value of 1 from right to left, and 'a' corresponds to the first bit value of 1.

```
gbase> SELECT MAKE_ SET(1,'a','b','c') FROM dual;
+--------------------------+
| MAKE_ SET(1,'a','b','c') |
+--------------------------+
| a                        |
+--------------------------+
1 row in set
```

Example 2: Performing an OR operation on 1 and 4 to obtain the bit value of 0101, and verifying it from right to left, that is, the bit values of the first and third bits are 1, and the str1 and str3 strings are returned.

```
gbase> SELECT MAKE_ SET(1 | 4,'hello','nice','world') FROM dual;
+---------------------------------------+
| MAKE_ SET(1 | 4,'hello','nice','world') |
+---------------------------------------+
| hello,world                           |
+---------------------------------------+
1 row in set
```

Example 3: NULL in a string column is not added to the result.

```
gbase> SELECT MAKE_ SET(1 | 4,'hello','nice',NULL,'world') FROM dual;
+--------------------------------------------+
| MAKE_ SET(1 | 4,'hello','nice',NULL,'world') |
+--------------------------------------------+
| hello                                      |
+--------------------------------------------+
```

```
1 row in set
```

Example 4: 0 does not have a corresponding str.

```
gbase> SELECT MAKE_ SET(0,'a','b','c') FROM dual;
+--------------------------+
| MAKE_ SET(0,'a','b','c') |
+--------------------------+
|                          |
+--------------------------+
1 row in set
```

5.1.5.3.26 MID(str,pos,len)

Function Description

Returns a substring of length len starting from the pos position in the string str. MID (str, pos, len) is equivalent to SUBSTRING (str, pos, len).

Example

Example 1: Substring() is equivalent to MID().

```
gbase> SELECT SUBSTRING('Quadratically',5,6),MID('Quadratically',5,6)
FROM dual;
+-------------------------------+--------------------------+
| SUBSTRING('Quadratically',5,6) | MID('Quadratically',5,6) |
+-------------------------------+--------------------------+
| ratica                        | ratica                   |
+-------------------------------+--------------------------+
1 row in set
```

5.1.5.3.27 NVL(string1,replace_with)

Function Description

If string1 is NULL, the NVL() function returns replace_ Otherwise, the value of string1 is returned.

Example

Example 1: If the value of the address column is NULL, return "UNKNOWN"; otherwise, return the value of address.

```
gbase> DROP TABLE IF EXISTS t_ user;
```

```
Query OK, 0 rows affected

gbase> CREATE TABLE t_ user (id int ,name varchar(10),address
varchar(200));
Query OK, 0 rows affected

gbase> INSERT INTO t_ user VALUES (1,'Tom','East
Street'),(2,'Mike',NULL),(3,'Rose','TANGREN ROAD'),(4,'White',NULL);
Query OK, 4 rows affected
Records: 4    Duplicates: 0    Warnings: 0

gbase> SELECT id,name,NVL(address,'UNKOWN') FROM t_ user;
+------+-------+-----------------------+
| id     | name    | NVL(address,'UNKOWN') |
+------+-------+-----------------------+
|       1 | Tom      | East Street                  |
|       2 | Mike    | UNKOWN                         |
|       3 | Rose    | TANGREN ROAD                |
|       4 | White | UNKOWN                         |
+------+-------+-----------------------+
4 rows in set
```

5.1.5.3.28 OCT(N)

Function Description

Returns a string of N's octal value. Here, N is a BIGINT type number. If N is a NULL, the return value is also NULL. OCT (N) is equivalent to CONV (N, 10,8).

Example

Example 1: Return the octal value of 12.

```
gbase> SELECT OCT(12) FROM dual;
+---------+
| OCT(12) |
+---------+
| 14          |
+---------+
1 row in set
```

Example 2: N is NULL and the return value is NULL.

```
gbase> SELECT OCT(NULL) FROM dual;
+-----------+
| OCT(NULL) |
```

```
+-----------+
| NULL      |
+-----------+
1 row in set
```

5.1.5.3.29 ORD(str)

Function Description

If the leftmost character of the string str is a multi byte character, the code for that character is returned. The code is calculated using the formula (1st byte code) +(2nd byte code × 256)+(3rd byte code ^{2562})...) Calculate the numerical value of its constituent bytes. If the leftmost character is not a multi byte character, the return value is the same as the return value of the ASCII () function.

Example

Example 1: str is "2" and returns the ASCII code value corresponding to 2.

```
gbase> SELECT ORD('2') FROM dual;
+----------+
| ORD('2') |
+----------+
|       50 |
+----------+
1 row in set
```

Example 2: str is "Nanjing University General" and returns the code corresponding to "South".

```
GBase>SELECT ORD ('Nanjing University General Motors') from dual;
+---------------------+
|ORD|
+---------------------+
|            15043991 |
+---------------------+
1 row in set
```

5.1.5.3.30 REPEAT(str,count)

Function Description

Returns a string consisting of a string str that has been repeated count times. If count<=0, returns an empty string; If str or count is NULL, the return value is NULL. The range of count values is bigint.

Example

Example 1: Returns the string after repeating 'GBase' 3 times.

```
gbase> SELECT REPEAT('GBase', 3) FROM dual;
+--------------------+
| REPEAT('GBase', 3) |
+--------------------+
| GBaseGBaseGBase    |
+--------------------+
1 row in set
```

 **be careful**

- If the number of bytes occupied by str * count>16M in repeat (str, count), please add max in the [gbased] column of the cluster configuration file_ allowed_ packet = 64M。 Because although repeat allows a single maximum tuple of 64M, the client defaults to max_ allowed_ If the packet is 16M, content larger than 16M will be intercepted and an error will be reported for processing.

5.1.5.3.31 REPLACE(str,from_str,to_str)

Function Description

The returned result is to convert all occurrences of 'from' in str_ Replace str with to_ The string after str.

Example

Example 1: Replace all occurrences of 'w' in 'www.gbase8a. com' with 'Ww'.

```
gbase> SELECT REPLACE('www.gbase8a.com', 'w', 'Ww') FROM dual;
+---------------------------------------+
| REPLACE('www.gbase8a.com', 'w', 'Ww') |
+---------------------------------------+
```

```
| WwWwWw.gbase8a.com                          |
+---------------------------------------------+
1 row in set
```

5.1.5.3.32 REVERSE(str)

Function Description

Returns a string output in reverse order.

Example

Example 1: Output "abc" in order from right to left.

```
gbase> SELECT REVERSE('abc') FROM dual;
+----------------+
| REVERSE('abc') |
+----------------+
| cba            |
+----------------+
1 row in set
```

5.1.5.3.33 RIGHT(str,len)

Function Description

Returns the len characters from the right side of the string str.

Example

Example 1: Returns the four rightmost characters of 'foobar'.

```
gbase> SELECT RIGHT('foobarbar', 4) FROM dual;
+-----------------------+
| RIGHT('foobarbar', 4) |
+-----------------------+
| rbar                  |
+-----------------------+
1 row in set
```

5.1.5.3.34 RPAD(str,len,padstr)

Function Description

Use the string padstr to fill the right side of str until its length reaches len characters, and then return the filled str. If the length of str is longer than len, it will be truncated to len characters.

Example

Example 1: Add "?" to the right of "hi" until the length is 5 digits.

```
gbase> SELECT RPAD('hi',5,'?') FROM dual;
+------------------+
| RPAD('hi',5,'?') |
+------------------+
| hi???            |
+------------------+
1 row in set
```

Example 2: If the length of 'hiacd' is greater than 2, it will be truncated to 2 characters.

```
gbase> SELECT RPAD('hiacd',2,'?') FROM dual;
+------------------+
| RPAD('hi',2,'?') |
+------------------+
| hi               |
+------------------+
1 row in set
```

5.1.5.3.35 RTRIM(str)

Function Description

Returns the string after removing the rightmost extra space from the string str.

Example

Example 1: Remove the extra space on the far right of 'bar'.

```
gbase> SELECT RTRIM('barbar     ') FROM dual;
+--------------------+
| RTRIM('barbar     ') |
+--------------------+
| barbar             |
+--------------------+
1 row in set
```

5.1.5.3.36 SUBSTRING()

Grammar format

```
SUBSTRING(str,pos)
SUBSTRING(str,pos,len)
```

Function Description

The SUBSTRING() function without the len parameter returns all remaining substrings from the pos position of the string str. The SUBSTRING() function with the len parameter returns a substring of len characters starting from the pos position of the string str. Pos can be a negative value. In this case, the starting position of the substring is the pos position from the end of the string forward. If len is a value less than 1, the return result is always an empty string.

Example

Example 1: Returns the substring of "Quadratically" starting from position 5.

```
gbase> SELECT SUBSTRING('Quadratically',5) FROM dual;
+-----------------------------+
| SUBSTRING('Quadratically',5) |
+-----------------------------+
| ratically                   |
+-----------------------------+
1 row in set
```

Example 2: Returns 6 characters starting from the 5th position of "Quadratically".

```
gbase> SELECT SUBSTRING('Quadratically',5,6) FROM dual;
+--------------------------------+
| SUBSTRING('Quadratically',5,6) |
+--------------------------------+
| ratica                         |
+--------------------------------+
1 row in set
```

Example 3: If pos is "-3", the returned substring is the 3 characters at the end of "Sakila".

```
gbase> SELECT SUBSTRING('Sakila', -3) FROM dual;
+-------------------------+
| SUBSTRING('Sakila', -3) |
+-------------------------+
| ila                     |
+-------------------------+
1 row in set
```

Example 4: If pos is "-5" and len is "3", the returned substring is "Sakila", which consists of 3 characters starting from the second position.

```
gbase> SELECT SUBSTRING('Sakila', -5, 3) FROM dual;
+----------------------------+
| SUBSTRING('Sakila', -5, 3) |
+----------------------------+
| aki                        |
+----------------------------+
1 row in set
```

5.1.5.3.37 SUBSTRING_ INDEX(str,delim,count)

Function Description

Returns the substring in the string str before the count delimiter delim. If count is a positive number, return all characters from the last (counting from the left) separator to the left; If count is a negative number, returns all characters from the last (counting from the right) separator to the right.

Example

Example 1: Count is a positive number that returns all characters from the separator '.' to the left.

```
gbase> SELECT SUBSTRING_ INDEX('www.gbase8a.com', '.', 2) FROM
dual;
+--------------------------------------------+
| SUBSTRING_ INDEX('www.gbase8a.com', '.', 2) |
+--------------------------------------------+
| www.gbase8a                                |
+--------------------------------------------+
1 row in set
```

Example 2: Count is a negative number that returns all characters from the separator '.' to the right.

```
gbase> SELECT SUBSTRING_ INDEX('www.gbase8a.com', '.', -2) FROM
dual;
+---------------------------------------------+
| SUBSTRING_ INDEX('www.gbase8a.com', '.', -2) |
+---------------------------------------------+
| gbase8a.com                                 |
+---------------------------------------------+
```

```
1 row in set
```

5.1.5.3.38 TO_ CHAR(number,[FORMAT])

Function Description

Convert the parameter number to a string and format the output. If the number of digits is greater than the formatting parameter FORMAT, the result will be displayed as' # '.

Parameter Description

Table 511 Parameter Description

Format parameters	meaning
,	Usually used as a grouping symbol, the number parameter is formatted as a string output in digital format, such as grouping by thousands, or grouping by hundreds or tens. Usually used in conjunction with 0, 9, and '.'. Example: 99999.
.	Format the number parameter as a decimal string output. Can only appear once. Usually used in conjunction with 0, 9, and. Example: 999.99.
$	The string converted to the currency meaning of US dollars can only appear at the beginning or end. Example: $999.
0	Placeholder, formatted number. If the number of digits in the parameter number is less than the formatted digits, 0 is displayed to fill in the digits. Note: 0 has a higher priority than 9. Example: 000.
nine	Placeholder, formatting number. Once the number of digits in the parameter number is less than the formatted digits, spaces are used to fill in the digits. Example: 999.
B、b	If the value of number is 0, replace it with a space and it can appear anywhere. Example: B9.99
EEEE、eeee	Output according to Scientific notation. Example: 9.99EEEE.
FM、fm	Remove spaces at the beginning and end of numbers. Example: FM909.9.
TME	Returns number according to Scientific notation.
X、x	Convert to hexadecimal. Each X represents a hexadecimal digit.

Format parameters	meaning
	For example, XX represents a two digit hexadecimal number. If number is converted to a hexadecimal number greater than X, then "#" is output. Note: The numerical value must be an integer greater than or equal to 0. The front can only be used in combination with 0 or FM.

Example

Example 1: Grouping by hundreds.

```
gbase> SELECT TO_ CHAR(987654321,'999,999,999') FROM dual;
+----------------------------------+
| TO_ CHAR(987654321,'999,999,999') |
+----------------------------------+
|   987,654,321                    |
+----------------------------------+
1 row in set
```

Example 2: Completing numerical digits with blank spaces.

```
gbase> SELECT TO_ CHAR(54321,'999,999,999') FROM dual;
+------------------------------+
| TO_ CHAR(54321,'999,999,999') |
+------------------------------+
|            54,321            |
+------------------------------+
1 row in set
```

Example 3: Because 0 has a higher priority than 9, both 100000 and million bits are displayed as 0, while tens of millions and billions of bits are displayed as spaces.

```
gbase> SELECT TO_ CHAR(54321,'990,999,999') FROM dual;
+------------------------------+
| TO_ CHAR(54321,'990,999,999') |
+------------------------------+
|       0,054,321              |
+------------------------------+
1 row in set

gbase> SELECT TO_ CHAR(-54321,'990,999,999') FROM dual;
+-------------------------------+
| TO_ CHAR(-54321,'990,999,999') |
+-------------------------------+
|     -0,054,321                |
```

```
+-------------------------------+
1 row in set
```

Example 4: Decimal formatting output.

```
gbase> SELECT TO_ CHAR(12.97,'099.99') FROM dual;
+-------------------------+
| TO_ CHAR(12.97,'099.99') |
+-------------------------+
|   012.97                |
+-------------------------+
1 row in set
```

Example 5: Decimal formatting output with three decimal places filled in.

```
gbase> SELECT TO_ CHAR(-12.97,'099.090') FROM dual;
+--------------------------+
| TO_ CHAR(-12.97,'099.090') |
+--------------------------+
| -012.970                 |
+--------------------------+
1 row in set

gbase> SELECT TO_ CHAR(12.97,'099.099') FROM dual;
+--------------------------+
| TO_ CHAR(12.97,'099.099') |
+--------------------------+
|   012.970                |
+--------------------------+
1 row in set
```

explain

Because the priority of 0 is higher than 9, whether it is 090 or 099, the output is formatted with 3 decimal places, and the filling positions are filled with 0.

Example 6: Format the output in US dollars, where the $formatting symbol can only appear in the first or last position.

```
gbase> SELECT TO_ CHAR(84.77,'$0099.99') FROM dual;
+--------------------------+
| TO_ CHAR(84.77,'$0099.99') |
+--------------------------+
|   $0084.77               |
```

```
+--------------------------+
1 row in set

gbase> SELECT TO_ CHAR(84.77,'0099.99$') FROM dual;
+--------------------------+
| TO_ CHAR(84.77,'0099.99$') |
+--------------------------+
|    $0084.77                          |
+--------------------------+
1 row in set
```

Example 7

The integer part is 0 and returns a space.

```
gbase> SELECT TO_ CHAR(0,'B00') FROM dual;
+------------------+
| TO_ CHAR(0,'B00') |
+------------------+
|                              |
+------------------+
1 row in set

gbase> SELECT HEX(TO_CHAR(0,'B00')) FROM dual;
+----------------------+
| HEX(TO_CHAR(0,'B00')) |
+----------------------+
| 202020                          |
+----------------------+
1 row in set
```

When the integer part is 1, it returns 01.

```
gbase> SELECT TO_ CHAR(1,'B00') FROM dual;
+------------------+
| TO_ CHAR(1,'B00') |
+------------------+
|    01                        |
+------------------+
1 row in set
```

When the integer part is 11, it returns 11.

```
gbase> SELECT TO_ CHAR(11,'B00') FROM dual;
+------------------+
```

```
| TO_ CHAR(11,'B00') |
+--------------------+
|    11              |
+--------------------+
1 row in set
```

Example 8: The value of FORMAT is "9.9EEEE". As it is a scientific calculation method, adding a 9 or 0 before the decimal place is sufficient. Multiple decimal places are meaningless.

```
gbase> SELECT TO_ CHAR(2008032001,'9.9EEEE') FROM dual;
+-------------------------------+
| TO_ CHAR(2008032001,'9.9EEEE') |
+-------------------------------+
|     2.0E+09                   |
+-------------------------------+
1 row in set

gbase> SELECT TO_ CHAR(2008032001,'c9.99EEEE') FROM dual;
+--------------------------------+
| TO_ CHAR(2008032001,'c9.99EEEE') |
+--------------------------------+
|     $2.01E+09                  |
+--------------------------------+
1 row in set
```

Example 9: The value of FORMAT is "FM90.9", remove the spaces at the beginning and end of "10.3".

```
gbase> SELECT TO_ CHAR(10.3,'FM90.9') FROM dual;
+-------------------------+
| TO_ CHAR(-10.3,'FM90.9') |
+-------------------------+
| 10.3                    |
+-------------------------+
1 row in set

gbase> SELECT TO_ CHAR(10.3,'90.9') FROM dual;
+----------------------+
| TO_ CHAR(10.3,'90.9') |
+----------------------+
|    10.3              |
+----------------------+
1 row in set
```

Example 10: The value of FORMAT is "TME".

```
gbase> SELECT TO_ CHAR(11,'TME') AS f_ SHOW FROM dual;
+---------+
| f_ SHOW    |
+---------+
| 1.1E+01 |
+---------+
1 row in set
```

Example 11: The value of FORMAT is "X", which returns the hexadecimal form of number.

```
gbase> SELECT TO_ CHAR(11,'X') FROM dual;
+-----------------+
| TO_ CHAR(11,'X') |
+-----------------+
|   B                        |
+-----------------+
1 row in set

gbase> SELECT TO_ CHAR(16,'XX') FROM dual;
+------------------+
| TO_ CHAR(16,'XX') |
+------------------+
|   10                       |
+------------------+
1 row in set
```

If number is converted to a hexadecimal number greater than X, output '#'.

```
gbase> SELECT TO_ CHAR(16,'X') FROM dual;
+-----------------+
| TO_ CHAR(16,'X') |
+-----------------+
| ##                        |
+-----------------+
1 row in set
```

5.1.5.3.39 TO_ CHAR(datetime,[FORMAT])

Function Description

Convert the parameter datetime to a string and format the output.

Parameter Description

Table -512 Parameter Description

Format parameters	meaning
, . ; :	In addition to the left standard, text is also allowed as a separator symbol. For example, the separator for month, month, day, and date. Used to format the output date.
AD	The abbreviation for the Latin word 'Anno Domini' means' AD 'and will be converted to' AD 'or' AD 'depending on the nls. If it is a date after AD, display AD. If it is a date from BC, display BC.
AM	Short for morning, the Chinese environment output is morning. If it is morning, return to AM. If it is in the afternoon, return to PM.
BC	Abbreviation for Latin Before Christ, which means that in BC, it will be converted to AD or BC according to different nls. If it is a date after AD, display AD. If it is a date from BC, display BC.
CC	Returns the century, represented in Arabic numerals.
D	On the day of the week, the serial number (1-7) is returned.
DAY	Returns the DAY part of the date. The returned form is in full English spelling, with the first letter capitalized.
DD	Same as DAY, but returns in numerical form (01-31).
DDD	The day in the date is the day of the year, and the returned numbers are 001 to 366.
DY	Same as DAY, but returns in English form, returning the first three letters. Capitalize the initial letter.
FF[n]	It is milliseconds, and if no numbers are added, the default precision is used, with a default precision of 6 bits. $1 \leq n \leq 9$。 Can only be used for timestamp types.
FM	Remove the spaces at the beginning and end of the date. Example: FM Month
FX	Fixed mode global option. Example: FX MONTH DD DAY
HH[12 | 24]	Represents an hour, defaults to a 12 hour system. HH12, 12 hour system. Return to (01-12). HH24, 24-hour system. Return to (00-23).
IW	The week of the year for ISO standards (1-52, or 1-53).
MI	Returns the number of minutes (00-59).
MM	Returns the month and Arabic numerals.
MON	Returns the month, which is abbreviated in English, with three English

Format parameters	meaning
	letters and the first letter in uppercase.
MONTH	Returning to the month, returning is English Pinyin. Capitalize the initial letter.
PM	The abbreviation for 'afternoon' is' afternoon 'in Chinese.
Q	Returns the quarter, with values ranging from 1 to 4.
RM	A month expressed in Roman numerals. Roman numerals are all capitalized.
RR or RRRR	Returns 2 or 4 digit years.
YY or YYYY	Returns 2 or 4 digit years.
SCC	Returns the century represented in numerical form.
SS	Returns seconds (0-59).
SSSSS	The cumulative number of seconds in a day starting from midnight (0-86399).
TS	Returns the time in hours, minutes, and seconds with AM or PM.
W	The algorithm is limited to the month to which the datetime parameter belongs in the week of the month.
WW	Same as IW.

Example

Example 1: Convert NOW() to the corresponding date format in FORMAT.

```
gbase> SELECT TO_ CHAR(NOW(),'YYYY/MM/DD') FROM dual;
+-----------------------------+
| TO_ CHAR(NOW(),'YYYY/MM/DD') |
+-----------------------------+
| 2020/04/01                  |
+-----------------------------+
1 row in set

gbase> SELECT TO_ CHAR(NOW(),'YYYY-MM-DD') FROM dual;
+-----------------------------+
| TO_ CHAR(NOW(),'YYYY-MM-DD') |
+-----------------------------+
| 2020-04-01                  |
+-----------------------------+
1 row in set

gbase> SELECT TO_ CHAR(NOW(),'YYYY,MM,DD') FROM dual;
+-----------------------------+
| TO_ CHAR(NOW(),'YYYY,MM,DD') |
+-----------------------------+
```

```
| 2020,04,01                    |
+-------------------------------+
1 row in set

gbase> SELECT TO_ CHAR(CURDATE(),'YYYY; MM; DD') FROM dual;
+-----------------------------------+
| TO_ CHAR(CURDATE(),'YYYY; MM; DD') |
+-----------------------------------+
| 2020; 04; 01                      |
+-----------------------------------+
1 row in set

gbase> SELECT TO_ CHAR (NOW, 'YYYY' year, MM, DD, day) from dual;
+----------------------------------------+
| TO_ CHAR (NOW, 'YYYY' MM 'DD' day)|
+----------------------------------------+
|April 1st, 2020|
+----------------------------------------+
1 row in set
```

Example 2: Convert CURDATE() to the corresponding date format in FORMAT.

```
gbase> SELECT TO_ CHAR(CURDATE(),'AD    YYYY-MM-DD') FROM
dual;
+-------------------------------------+
| TO_ CHAR(CURDATE(),'AD    YYYY-MM-DD') |
+-------------------------------------+
| AD    2020-04-01                    |
+-------------------------------------+
1 row in set
```

Example 3: Comparing NOW () and converting NOW () to "AM HH12: MI: SS" format.

```
gbase> SELECT NOW(),TO_ CHAR(NOW(),'AM HH12:MI:SS') FROM
dual;
+---------------------+--------------------------------+
| NOW()               | TO_ CHAR(NOW(),'AM HH12:MI:SS') |
+---------------------+--------------------------------+
| 2020-04-01 16:35:43 | PM 04:35:43                    |
+---------------------+--------------------------------+
1 row in set

gbase> SELECT NOW(),TO_
CHAR(TIMESTAMPADD(HOUR,8,NOW()),'AM HH12:MI:SS') AS f_
FormatShow FROM dual;
```

```
+---------------------+--------------+
| NOW()               | f_ FormatShow |
+---------------------+--------------+
| 2020-04-01 16:36:03 | AM 12:36:03  |
+---------------------+--------------+
1 row in set
```

Example 4: Returns the century of CURDATE().

```
gbase> SELECT TO_ CHAR(CURDATE(),'CC') FROM dual;
+-------------------------+
| TO_ CHAR(CURDATE(),'CC') |
+-------------------------+
| 21                      |
+-------------------------+
1 row in set
```

Example 5: The system defaults to Sunday as the first day of the week, and "2020 04 01" is Wednesday.

The value of FORMAT is "D" and the return value is 6.

```
gbase> SELECT CURDATE(),TO_ CHAR(CURDATE(),'D') FROM dual;
+------------+------------------------+
| CURDATE()  | TO_ CHAR(CURDATE(),'D') |
+------------+------------------------+
| 2020-04-01 | 4                      |
+------------+------------------------+
1 row in set
```

The value of FORMAT is "DAY", which returns the full English spelling of Wednesday, with the first letter capitalized.

```
gbase> SELECT CURDATE(),TO_ CHAR(CURDATE(),'DAY') FROM dual;
+--------------------+----------------------------------+
| date('2020-04-01') | TO_ CHAR(date('2020-04-01'),'DAY') |
+--------------------+----------------------------------+
| 2020-04-01         | Wednesday                        |
+--------------------+----------------------------------+
1 row in set
```

The value of FORMAT is "DD" and the return value is 11.

```
gbase> SELECT CURDATE(),TO_ CHAR(CURDATE(),'DD') FROM dual;
+------------+------------------------+
```

```
| CURDATE()    | TO_ CHAR(CURDATE(),'DD') |
+------------+-------------------------+
| 2020-04-01 | 01                      |
+------------+-------------------------+
1 row in set
```

The value of FORMAT is "DDD", and the return "2020 04 01" indicates the day of 2020.

```
gbase> SELECT CURDATE(),TO_ CHAR(CURDATE(),'DDD') FROM dual;
+------------+-------------------------+
| CURDATE()    | TO_ CHAR(CURDATE(),'DDD') |
+------------+-------------------------+
| 2020-04-01 | 092                     |
+------------+-------------------------+
1 row in set
```

The value of FORMAT is "DY", which returns the first three letters of Wednesday in English, with the first letter capitalized.

```
gbase> SELECT CURDATE(),TO_ CHAR(CURDATE(),'DY') FROM dual;
+------------+-------------------------+
| CURDATE()    | TO_ CHAR(CURDATE(),'DY') |
+------------+-------------------------+
| 2020-04-01 | Wed                     |
+------------+-------------------------+
1 row in set
```

Example 6: Query the milliseconds of the current time, default to 6 bits.

```
gbase> SELECT CURRENT_ TIMESTAMP(),TO_
CHAR(CURRENT_TIMESTAMP(),'FF') FROM dual;
+--------------------+----------------------------------+
| CURRENT_ TIMESTAMP() | TO_ CHAR(CURRENT_TIMESTAMP(),'FF') |
+--------------------+----------------------------------+
| 2020-04-01 16:37:51 | 000000                          |
+--------------------+----------------------------------+
1 row in set

gbase> SELECT CURRENT_ TIMESTAMP(),TO_
CHAR(CURRENT_TIMESTAMP(),'FF9') FROM dual;
+--------------------+----------------------------------+
| CURRENT_ TIMESTAMP() | TO_ CHAR(CURRENT_TIMESTAMP(),'FF9') |
+--------------------+----------------------------------+
| 2020-04-01 16:38:15 | 000000000                       |
```

```
+---------------------+-----------------------------------+
1 row in set
```

Example 7: FORMAT is either "FX" or "FM".

```
gbase> SELECT CURRENT_ TIMESTAMP(),TO_
CHAR(CURRENT_TIMESTAMP(),'FX YYYY-MM-DD') FROM dual;
+---------------------+-----------------------------------------------+
| CURRENT_ TIMESTAMP() | TO_ CHAR(CURRENT_TIMESTAMP(),'FX
YYYY-MM-DD') |
+---------------------+-----------------------------------------------+
| 2020-04-01 16:38:39 |    2020-04-01                                 |
+---------------------+-----------------------------------------------+
1 row in set

gbase> SELECT CURRENT_ TIMESTAMP(),TO_
CHAR(CURRENT_TIMESTAMP(),'FM YYYY-MM-DD') FROM dual;
+---------------------+-----------------------------------------------+
| CURRENT_ TIMESTAMP() | TO_ CHAR(CURRENT_TIMESTAMP(),'FM
YYYY-MM-DD') |
+---------------------+-----------------------------------------------+
| 2020-04-01 16:39:12 |    2020-04-01                                 |
+---------------------+-----------------------------------------------+
1 row in set
```

Example 8: FORMAT is "HH" and returns the hour.

```
gbase> SELECT CURRENT_ TIMESTAMP(),TO_
CHAR(CURRENT_TIMESTAMP(),'HH') FROM dual;
+---------------------+-----------------------------------+
| CURRENT_ TIMESTAMP() | TO_ CHAR(CURRENT_TIMESTAMP(),'HH') |
+---------------------+-----------------------------------+
| 2021-06-03 15:00:58 | 03                                |
+---------------------+-----------------------------------+
1 row in set (Elapsed: 00:00:00.01)

FORMAT is "HH12" and returns the hours in 12 hour format.

gbase> SELECT NOW(),TO_ CHAR(NOW(),'HH12') FROM dual;
+---------------------+----------------------+
| NOW()               | TO_ CHAR(NOW(),'HH12') |
+---------------------+----------------------+
| 2021-06-03 15:02:00 | 03                   |
+---------------------+----------------------+
1 row in set (Elapsed: 00:00:00.02)
```

```
FORMAT is "HH24" and returns the hours in 24-hour format.

gbase> SELECT NOW(),TO_
CHAR(TIMESTAMPADD(HOUR,8,NOW()),'HH24') FROM dual;
+---------------------+-------------------------------------------+
| NOW()               | TO_
CHAR(TIMESTAMPADD(HOUR,8,NOW()),'HH24') |
+---------------------+-------------------------------------------+
| 2021-06-03 15:03:43 | 23                                        |
+---------------------+-------------------------------------------+
1 row in set
```

Example 9: FORMAT is "IW", which returns the week ordinal of the year.

```
gbase> SELECT TO_ CHAR(NOW(),'IW') FROM dual;
+---------------------+
| TO_ CHAR(NOW(),'IW') |
+---------------------+
| 14                  |
+---------------------+
1 row in set
```

Example 10: FORMAT is "MI" and returns the number of minutes.

```
gbase> SELECT NOW(),TO_ CHAR(NOW(),'MI') FROM dual;
+---------------------+---------------------+
| NOW()               | TO_ CHAR(NOW(),'MI') |
+---------------------+---------------------+
| 2020-04-01 17:00:33 | 00                  |
+---------------------+---------------------+
1 row in set
```

Example 11: FORMAT is "MM", "MON", "MONTH", and returns the month in different forms.

```
gbase> SELECT NOW(),TO_ CHAR(NOW(),'MM') FROM dual;
+---------------------+---------------------+
| NOW()               | TO_ CHAR(NOW(),'MM') |
+---------------------+---------------------+
| 2020-04-01 17:01:09 | 04                  |
+---------------------+---------------------+
1 row in set

gbase> SELECT NOW(),TO_ CHAR(NOW(),'MON') FROM dual;
```

```
+---------------------+----------------------+
| NOW()               | TO_ CHAR(NOW(),'MON') |
+---------------------+----------------------+
| 2020-04-01 17:01:31 | Apr                  |
+---------------------+----------------------+
1 row in set

gbase> SELECT NOW(),TO_ CHAR(NOW(),'MONTH') FROM dual;
+---------------------+-----------------------+
| NOW()               | TO_ CHAR(NOW(),'MONTH') |
+---------------------+-----------------------+
| 2020-04-01 09:44:50 | April                 |
+---------------------+-----------------------+
1 row in set
```

Example 12: The format is "PM HH12: MI: SS".

```
gbase> SELECT NOW(),TO_ CHAR(NOW(),'PM HH12:MI:SS') FROM
dual;
+---------------------+-------------------------------+
| NOW()               | TO_ CHAR(NOW(),'PM HH12:MI:SS') |
+---------------------+-------------------------------+
| 2020-04-01 17:06:21 | PM 05:06:21                   |
+---------------------+-------------------------------+
1 row in set

gbase> SELECT NOW(),TO_
CHAR(TIMESTAMPADD(HOUR,8,NOW()),'PM HH12:MI:SS') AS f_
FormatShow FROM dual;
+---------------------+--------------+
| NOW()               | f_ FormatShow |
+---------------------+--------------+
| 2020-04-01 10:40:45 | PM 06:40:45  |
+---------------------+--------------+
1 row in set
```

Example 13: The format is "Q YYYY-MM-DD", and the date in NOW() is the quarter.

```
gbase> SELECT NOW(),TO_ CHAR(NOW(),'Q YYYY-MM-DD') FROM
dual;
+---------------------+-------------------------------+
| NOW()               | TO_ CHAR(NOW(),'Q YYYY-MM-DD') |
+---------------------+-------------------------------+
| 2020-04-01 17:07:00 | 2 2020-04-01                  |
+---------------------+-------------------------------+
```

```
1 row in set
```

Example 14: If FORMAT is "RM", return the month represented by Roman numerals.

```
gbase> SELECT NOW(),TO_ CHAR(NOW(),'RM') FROM dual;
+---------------------+---------------------+
| NOW()               | TO_ CHAR(NOW(),'RM') |
+---------------------+---------------------+
| 2020-04-01 16:46:52 | IV                  |
+---------------------+---------------------+
1 row in set
```

Example 15: FORMAT is "RR" or "RRRR", returns a year in 2 or 4 digits.

```
gbase> SELECT TO_ CHAR(NOW(),'RR') FROM dual;
+--------------------+
| TO_ CHAR(NOW(),'RR') |
+--------------------+
| 20                 |
+--------------------+
1 row in set

gbase> SELECT TO_ CHAR(NOW(),'RRRR') FROM dual;
+----------------------+
| TO_ CHAR(NOW(),'RRRR') |
+----------------------+
| 2020                 |
+----------------------+
1 row in set

gbase> SELECT TO_
CHAR(TIMESTAMPADD(YEAR,-1200,NOW()),'RRRR') FROM dual;
+-----------------------------------------------+
| TO_ CHAR(TIMESTAMPADD(YEAR,-1200,NOW()),'RRRR') |
+-----------------------------------------------+
| 0820                                          |
+-----------------------------------------------+
1 row in set
```

Example 16: FORMAT is "SCC" and returns the number of centuries to which the date belongs.

```
gbase> SELECT NOW(),TO_ CHAR(NOW(),'SCC') FROM dual;
+---------------------+---------------------+
| NOW()               | TO_ CHAR(NOW(),'SCC') |
```

```
+---------------------+----------------------+
| 2020-04-01 17:08:12 |    21                |
+---------------------+----------------------+
1 row in set

gbase> SELECT TIMESTAMPADD(YEAR,-20,NOW()) AS f_
DATETIME,TO_ CHAR(TIMESTAMPADD(YEAR,-20,NOW()),'SCC') AS
f_ AD FROM dual;
+---------------------+------+
| f_ DATETIME         | f_ AD |
+---------------------+------+
| 2000-04-01 17:08:26 |    20    |
+---------------------+------+
1 row in set
```

Example 17: FORMAT is "SSSSS", which returns the cumulative number of seconds in a day starting from midnight.

```
gbase> SELECT NOW(),TO_ CHAR(NOW(),'SS'),TO_
CHAR(NOW(),'SSSSS') FROM dual;
+---------------------+---------------------+-----------------------+
| NOW()               | TO_ CHAR(NOW(),'SS') | TO_
CHAR(NOW(),'SSSSS') |
+---------------------+---------------------+-----------------------+
| 2020-04-01 17:08:40 | 40                  | 61720                 |
+---------------------+---------------------+-----------------------+
1 row in set
```

Example 18: FORMAT is "TS" and returns time in hours, minutes, and seconds with AM or PM.

```
gbase> SELECT NOW(),TO_ CHAR(NOW(),'TS') FROM dual;
+---------------------+---------------------+
| NOW()               | TO_ CHAR(NOW(),'TS') |
+---------------------+---------------------+
| 2020-04-01 17:11:38 | 05:11:38 PM         |
+---------------------+---------------------+
1 row in set

gbase> SELECT TIMESTAMPADD(HOUR,8,NOW()) AS f_ now,TO_
CHAR(TIMESTAMPADD(HOUR,8,NOW()),'TS') AS f_ now_ ts FROM
dual;
+---------------------+-------------+
| f_ now              | f_ now_ ts  |
+---------------------+-------------+
```

```
| 2020-04-02 01:11:48 | 01:11:48 AM |
+---------------------+-------------+
1 row in set
```

Example 19: FORMAT is "W" and returns the week ordinal of the month in which the date is located.

```
gbase> SELECT NOW(),TO_ CHAR(NOW(),'W') FROM dual;
+---------------------+--------------------+
| NOW()               | TO_ CHAR(NOW(),'W') |
+---------------------+--------------------+
| 2020-04-01 17:12:14 | 1                  |
+---------------------+--------------------+
1 row in set
```

5.1.5.3.40 TO_ NUMBER(expr)

Function Description

Convert the data contained in the string expr into NUMBER type data. The form of expr can be any string that supports formatting, such as "111.0023" or "23000000".

Example

Example 1: expr is "-12.340000".

```
gbase> SELECT TO_ NUMBER('-12.340000') FROM dual;
+-------------------------+
| TO_ NUMBER('-12.340000') |
+-------------------------+
|                  -12.34 |
+-------------------------+
1 row in set
```

Example 2: expr is "12.34".

```
gbase> SELECT TO_ NUMBER('12.34') FROM dual;
+--------------------+
| TO_ NUMBER('12.34') |
+--------------------+
|              12.34 |
+--------------------+
1 row in set
```

Example 3: expr is "+00000123".

```
gbase> SELECT TO_ NUMBER('+000000123') FROM dual;
+-------------------------+
| TO_ NUMBER('+000000123') |
+-------------------------+
|                      123 |
+-------------------------+
1 row in set
```

Example 4: expr is "1234".

```
gbase> SELECT TO_ NUMBER('1234') FROM dual;
+-------------------+
| TO_ NUMBER('1234') |
+-------------------+
|              1234 |
+-------------------+
1 row in set
```

5.1.5.3.41 TRANSLATE(char,from_string,to_string)

Function Description

From contained in char_ Replace the string character with 'to'_ String, and then return the replaced string.

- to_ String cannot be omitted.
- If from_ String ratio to_ If the string is long, then in 'from'_ In string instead of in to_ The additional characters in the string will be removed from the char because they do not have corresponding replacement characters.
- If any parameter in TRANSLATE is NULL, the result is also NULL.

Example

Example 1: from_ String length is longer than to_ String, from_ In string instead of in to_ Extra characters in string will be removed from char.

```
gbase> SELECT TRANSLATE('123abc','2dc','4e') FROM dual;
+---------------------------------+
| TRANSLATE('123abc','2dc','4e') |
+---------------------------------+
| 143ab                           |
+---------------------------------+
1 row in set
```

explain

Because from_ String and to_ The positions of strings correspond one-to-one, with 2 corresponding to 4 and d corresponding to e.

There is no corresponding value for c, so c will be deleted. The 2 in the character will be replaced with 4, d will not be replaced because there is no corresponding replacement character in the string, and c will be removed from the string because there is no corresponding replacement character. Therefore, the output result is 143ab.

Example 2: from_ String length is longer than to_ String, from_ In string instead of in to_ Extra characters in string will be removed from char.

```
gbase> SELECT TRANSLATE('13579abc','13a','24') FROM dual;
+-----------------------------------+
| TRANSLATE('13579abc','13a','24') |
+-----------------------------------+
| 24579bc                           |
+-----------------------------------+
1 row in set
```

Example 3: from_ The string is NULL, and the return value is NULL.

```
gbase> SELECT TRANSLATE('23',NULL,'a') FROM dual;
+--------------------------+
| TRANSLATE('23',NULL,'a') |
+--------------------------+
| NULL                     |
+--------------------------+
1 row in set
```

5.1.5.3.42 TRIM

Function Description

TRIM([{BOTH | LEADING | TRAILING} [trim_char] FROM] str)。 Remove all trims from the string str_ Char prefix or suffix, and then return it. If no BOTH, LEADING, or TRAILING modifiers are given, BOTH will be assumed. If trim is not specified_ Char, spaces will be removed.

Example

Example 1: No trim specified_ Char, spaces will be removed.

```
gbase> SELECT TRIM('    bar      ') FROM dual;
+------------------+
| TRIM('    bar      ') |
+------------------+
| bar                        |
+------------------+
1 row in set
```

Example 2: Using the LEADING modifier.

```
gbase> SELECT TRIM(LEADING 'x' FROM 'xxxbarxxx') FROM dual;
+-------------------------------------+
| TRIM(LEADING 'x' FROM 'xxxbarxxx') |
+-------------------------------------+
| barxxx                                          |
+-------------------------------------+
1 row in set
```

Example 3: Using the BOTH modifier.

```
gbase> SELECT TRIM(BOTH 'x' FROM 'xxxbarxxx') FROM dual;
+---------------------------------+
| TRIM(BOTH 'x' FROM 'xxxbarxxx') |
+---------------------------------+
| bar                                        |
+---------------------------------+
1 row in set
```

Example 4: Using the TRAILING modifier.

```
gbase> SELECT TRIM(TRAILING 'xyz' FROM 'barxxyz') FROM dual;
+-------------------------------------+
| TRIM(TRAILING 'xyz' FROM 'barxxyz') |
+-------------------------------------+
| barx                                           |
+-------------------------------------+
1 row in set
```

5.1.5.3.43 UCASE(str)

Function Description

Set the mapping according to the current character set, change all characters in the string str to uppercase, and then return the value.

Example

Example 1: Convert a string to uppercase.

```
gbase> SELECT UCASE('Gbase') FROM dual;
+----------------+
| UCASE('Gbase') |
+----------------+
| GBASE          |
+----------------+
1 row in set
```

5.1.5.3.44 UNHEX(str)

Function Description

The inverse operation of HEX (str). It interprets each pair of hexadecimal digits in the parameter as a numerical value, which is then converted into a character represented by a numerical value. The returned result character is a binary character.

Example

Example 1: Convert str into numeric characters, where str is a hexadecimal digit.

```
gbase> SELECT UNHEX('4742617365') FROM dual;
+---------------------+
| UNHEX('4742617365') |
+---------------------+
| GBase               |
+---------------------+
1 row in set
```

Example 2: Convert str into numeric characters, where str is a hexadecimal digit.

```
gbase> SELECT 0x4742617365 FROM dual;
+--------------+
```

```
| 0x4742617365 |
+--------------+
| GBase        |
+--------------+
1 row in set
```

Example 3: UNHEX (HEX()) function.

```
gbase> SELECT UNHEX(HEX('string')) FROM dual;
+----------------------+
| UNHEX(HEX('string')) |
+----------------------+
| string               |
+----------------------+
1 row in set
```

Example 4: HEX (UNHEX()) function.

```
gbase> SELECT HEX(UNHEX('1267')) FROM dual;
+--------------------+
| HEX(UNHEX('1267')) |
+--------------------+
| 1267               |
+--------------------+
1 row in set
```

5.1.5.3.45 UPPER(str)

Function Description

Set the mapping according to the current character set, change all characters in the string str to uppercase, and then return the value. UPPER() is equivalent to UCASE().

Example

Example 1: Convert a string to uppercase.

```
gbase> SELECT UPPER('Hej') FROM dual;
+--------------+
| UPPER('Hej') |
+--------------+
| HEJ          |
+--------------+
```

```
1 row in set
```

5.1.5.3.46 String conversion type function

Function Description

GBase 8a MPP Cluster will automatically convert numbers to strings or convert strings to numbers. If a binary string is passed as a parameter to a string function, the result return is also a binary string. A number is converted into a string, which is considered a binary string but may affect the final result.

Example

Example 1: Automatically convert numbers to strings, or convert strings to numbers.

```
gbase> SELECT 1 + '1' FROM dual;
+---------+
| 1 + '1' |
+---------+
|       2 |
+---------+
1 row in set

gbase> SELECT CONCAT(2,' test') FROM dual;
+-------------------+
| CONCAT(2,' test') |
+-------------------+
| 2 test            |
+-------------------+
1 row in set

gbase> SELECT 38.8, CONCAT(38.8) FROM dual;
+------+--------------+
| 38.8 | CONCAT(38.8) |
+------+--------------+
| 38.8 | 38.8         |
+------+--------------+
1 row in set
```

Example 2: If it is explicitly necessary to convert a number into a string, the CAST() or CONCAT() functions can be used. Suggest using CAST().

```
gbase> SELECT 38.8, CAST(38.8 AS CHAR) FROM dual;
```

```
+------+--------------------+
| 38.8 | CAST(38.8 AS CHAR) |
+------+--------------------+
| 38.8 | 38.8               |
+------+--------------------+
1 row in set
```

5.1.5.3.47 expr LIKE pat [ESCAPE 'escape-char']

Function Description

expr LIKE pat [ESCAPE 'escape-char']。 Pattern matching using simple regular expressions in SQL for comparison. If the expression expr matches pat, it returns 1 (TRUE), otherwise it returns 0 (FALSE). A pattern may not necessarily be a literal string, for example, it can use string expressions or table columns. The two wildcards shown below can be used in conjunction with LIKE in the pattern.

Table 513 Wildcard Description

character	meaning
%	Match any multiple characters or zero characters.
_	Strictly match a character.

Example

Example 1: If expr matches pat and the wildcard is' _ ', it returns 1.

```
gbase> SELECT 'David!' LIKE 'David_' FROM dual;
+-----------------------+
| 'David!' LIKE 'David_' |
+-----------------------+
|                     1 |
+-----------------------+
1 row in set
```

Example 2: If expr matches pat and the wildcard is'% ', it returns 1.

```
gbase> SELECT 'David!' LIKE '%D%v%' FROM dual;
+-----------------------+
| 'David!' LIKE '%D%v%' |
+-----------------------+
|                     1 |
+-----------------------+
1 row in set
```

explain

The position of substr in str, counted starting with 1.

The returned positions are counted in the forward direction of the entire string, regardless of the starting position.

Table 514 String Description

character string	meaning
\%	Match a% character.
_	Match a_ Characters.

Example 3: expr does not match pat, returns 0.

```
gbase> SELECT 'David!' LIKE 'David\_' FROM dual;
+-------------------------+
| 'David!' LIKE 'David\_' |
+-------------------------+
|                       0 |
+-------------------------+
1 row in set
```

Example 4: The escape character " _" matches "_".

```
gbase> SELECT 'David_' LIKE 'David\_' FROM dual;
+-------------------------+
| 'David_' LIKE 'David\_' |
+-------------------------+
|                       1 |
+-------------------------+
1 row in set
```

Example 5: To specify a different escape character, you can use the ESCAPE clause.

```
gbase> SELECT 'David_' LIKE 'David|_' ESCAPE '|' FROM dual;
+-----------------------------------+
| 'David_' LIKE 'David|_' ESCAPE '|' |
+-----------------------------------+
|                                 1 |
+-----------------------------------+
1 row in set
```

Example 6

```
gbase> SELECT 'abc' LIKE 'ABC' FROM dual;
+------------------+
| 'abc' LIKE 'ABC' |
+------------------+
|                1 |
+------------------+
1 row in set
```

Example 7

```
gbase> SELECT 'abc' LIKE BINARY 'ABC' FROM dual;
+--------------------------+
| 'abc' LIKE BINARY 'ABC' |
+--------------------------+
|                        0 |
+--------------------------+
1 row in set
```

explain

The above examples 6 and 7 indicate that string comparison ignores case, unless either operand is a binary string.

Example 8: LIKE is allowed to be used on a numeric expression.

```
gbase> SELECT 10 LIKE '1%' FROM dual;
+--------------+
| 10 LIKE '1%' |
+--------------+
|            1 |
+--------------+
1 row in set
```

 **be careful**

- Due to GBase 8a MPP Cluster using C escape syntax in strings (for example, using " n" to represent a newline character), in LIKE strings, the "" used must be double written;
- For example, to find ' n', it must be written as' n '. To find ", it must be written as' '. The reason is that the backslash symbol will be stripped once by the parsing program, and during pattern matching, it will be stripped again, leaving only one backslash symbol to accept matching.

5.1.5.3.48 expr NOT LIKE pat [ESCAPE 'escape-char']

Function Description

Expr NOT LIKE pat [ESCAPE 'escape char'] is equivalent to NOT (expr LIKE pat [E SCAPE 'escape char']); If the expression expr matches pat, return 0; Otherwise, it returns 1.

Example

Example 1: Matching expr with pat returns 0 for matching and 1 for mismatch.

```
gbase> SELECT 'David_' NOT LIKE 'David|_' ESCAPE '|';
+-----------------------------------------+
| 'David_' NOT LIKE 'David|_' ESCAPE '|' |
+-----------------------------------------+
|                                       0 |
+-----------------------------------------+
1 row in set
```

5.1.5.3.49 expr REGEXP pat, expr RLIKE pat

Function Description

Perform a pattern comparison on the string expression expr according to the pattern pat. A pattern can be an extended regular expression, a literal string, a string expression, or a table column. If the expression expr matches pat, it returns 1; otherwise, it returns 0. RLIKE is a synonym for REGEXP. REGEXP is case insensitive to normal (not binary) strings.

Example

Example 1: expr does not match pat, returns 0.

```
gbase> SELECT 'Monty!' REGEXP 'm%y%%' FROM dual;
+-------------------------+
| 'Monty!' REGEXP 'm%y%%' |
+-------------------------+
|                       0 |
+-------------------------+
1 row in set
```

Example 2: If expr matches pat, it returns 1.

```
gbase> SELECT 'Monty!' REGEXP '.*' FROM dual;
+----------------------+
| 'Monty!' REGEXP '.*' |
+----------------------+
|                    1 |
+----------------------+
1 row in set
```

Example 3: If expr matches pat, it returns 1. The expression contains escape character.

```
gbase> SELECT 'new*\n*line' REGEXP 'new\\*.\\* line' FROM dual;
+---------------------------------------+
| 'new*\n*line' REGEXP 'new\\*.\\* line' |
+---------------------------------------+
|                                     1 |
+---------------------------------------+
1 row in set
```

Example 4: Adding BINARY before an expression is case sensitive.

```
gbase> SELECT 'a' REGEXP 'A', 'a' REGEXP BINARY 'A' FROM dual;
+----------------+-----------------------+
| 'a' REGEXP 'A' | 'a' REGEXP BINARY 'A' |
+----------------+-----------------------+
|              1 |                     0 |
+----------------+-----------------------+
1 row in set
```

Example 5: If expr matches pat, it returns 1.

```
gbase> SELECT 'a' REGEXP '^[a-d]' FROM dual;
```

```
+---------------------+
| 'a' REGEXP '^[a-d]' |
+---------------------+
|                   1 |
+---------------------+
1 row in set
```

5.1.5.3.50 expr NOT REGEXP pat, expr NOT RLIKE pat

Function Description

Expr NOT REGEXP pat, expr NOT RLIKE pat is equivalent to NOT (expr REGEXP pat). If the expression expr matches pat, it returns 0; otherwise, it returns 1. If expr or pat is NULL, then the result is NULL.

Example

Example 1: expr does not match pat, returns 1.

```
gbase>    SELECT NOT ('Monty!' REGEXP 'm%y%%' ) FROM dual;
+--------------------------------+
| not ('Monty!' REGEXP 'm%y%%' ) |
+--------------------------------+
|                              1 |
+--------------------------------+
1 row in set
```

Example 2: If expr or pat is NULL, the result is also NULL.

```
gbase> SELECT NULL REGEXP 'David_', ' David!' REGEXP NULL FROM
dual;
+----------------------+-----------------------+
| NULL REGEXP 'David_' | ' David!' REGEXP NULL    |
+----------------------+-----------------------+
|                 NULL |                  NULL |
+----------------------+-----------------------+
1 row in set
```

5.1.5.3.51 STRCMP(expr1,expr2)

Function Description

STRCMP() is a string comparison function. If the strings expr1 and expr2 are the same, STRCMP() returns 0. If expr1 is less than expr2 based on the current sorting order,

return -1; otherwise, return 1.

Example

Example 1: expr1 returns -1 based on the current sorting order being less than expr2.

```
gbase> SELECT STRCMP('text', 'text2') FROM dual;
+-------------------------+
| STRCMP('text', 'text2') |
+-------------------------+
|                      -1 |
+-------------------------+
1 row in set
```

Example 2: expr1 returns 1 based on the current sorting order being greater than expr2.

```
gbase> SELECT STRCMP('text2', 'text') FROM dual;
+-------------------------+
| STRCMP('text2', 'text') |
+-------------------------+
|                       1 |
+-------------------------+
1 row in set
```

Example 3: expr1 returns 0 based on the current sorting order being equal to expr2.

```
gbase> SELECT STRCMP('text', 'text') FROM dual;
+-------------------------+
| STRCMP('text', 'text') |
+-------------------------+
|                       0 |
+-------------------------+
1 row in set
```

explain

When comparing, STRCMP() uses the current character set, which makes the default comparison behavior case insensitive unless any or all of the operands are binary strings.

5.1.5.3.52 SPLIT_ PART

Function Description

SPLIT_ The PART function supports string cutting in both multi byte and single byte encoding scenarios, and can be divided into the following situations based o n the field value and the current number of string separators n:

- When 0<field<=n+1, returns the field segment of the string from left to ri ght.
- When field=0, str length is 0, or delimiter length is 0, an empty string is returned.
- When field>n+1, returns an empty string.
- When n=0, it means that the string does not have a specified separator, a nd regardless of the value of field, it returns an empty string.
- When - (n+1)<=field<0, returns the field segment of the string from right to left.
- When field<- (n+1), returns an empty string.
- Returns NULL when at least one of the three parameters of the function is null.
- When the function parameter is less than three, an error will be reported.

Grammar format

SPLIT_ PART(*str* string,*delimiter* string,*field* int)

Table 597 Parameter Description

Parameter Name	explain
str	It supports Chinese, English, and all symbols in the ascii table, such as symbols, special symbols, and mathematical notation. Support for col (numerical type, character type, date and time type) is currently not supported for properties listed as binary data types blob or longblob. Support parameters as functions. Can be null.
delimiter	It supports Chinese, English, and all symbols in ascii table, such

Parameter Name	explain
	as special symbols, mathematical notation, and does not support the use of separate escape symbols' ' It can be single or multi character. Support parameters as functions. Can be null.
field	Value range: The field value is a signed int type [-21474836482147483647], which can be null.

Example

Example 1: String cutting is performed based on the field value and the number of current string separators n.

```
SELECT SPLIT_ PART('www.gbase.com.cn','.','1');

+-------------------------------------------------+
| SPLIT_ PART('www.gbase.com.cn','.','1') |
+-------------------------------------------------+
| www                                             |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

SELECT SPLIT_ PART('www.gbase.com.cn','.',9999);

+---------------------------------------------------------+
| SPLIT_ PART('www.gbase.com.cn','.','9999') |
+---------------------------------------------------------+
|                                                         |
+---------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
SELECT SPLIT_ PART('www.gbase.com.cn','.',2);
+-------------------------------------------------+
| SPLIT_ PART('www.gbase.com.cn','.',2) |
+-------------------------------------------------+
| gbase                                           |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

SELECT SPLIT_ PART('www.gbase.com.cn','.',-1);
```

```
+-------------------------------------------------+
| SPLIT_ PART('www.gbase.com.cn','.',-1) |
+-------------------------------------------------+
| cn                                              |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.01)

SELECT SPLIT_ PART ('nan, da, tong, use ',. ', -2);
+-----------------------------------------------+
| SPLIT_ PART|
+-----------------------------------------------+
|Communication|
+-----------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

SELECT SPLIT_ PART ('nan, da, tong, use ',. ', null);
+-----------------------------------------------+
| SPLIT_ PART|
+-----------------------------------------------+
| NULL                                          |
+-----------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.3.53 INITCAP(expr)

Function Description

The function initcap (expr) normalizes the parameter string, converting the first letter of each word to uppercase and non first letter to lowercase.

Grammar format

```
INITCAP(expr);
```

Table 597 Parameter Description

Parameter Name	explain
expr	It can be a string, a column, a function, a constant, or a subquery with only a single column projection.

explain

The letters in this function refer to unicode letters, which not only include English letters and Arabic numerals, but also include Chinese characters, pinyin, Japanese hiragana, katakana, Western European letters, and other letters and numbers from various countries. If these letters are located at the beginning of the current word and are lowercase letters with corresponding uppercase letters, they are converted to uppercase letters.

If the current letter is a non initial letter of the current word, determine if there is a corresponding lowercase letter, and if so, convert it to lowercase.

The method for determining letters and non letters in a function is fixed.

Method of changing or specifying letter ranges and separators without relying on parameter passing

When comparing, STRCMP() uses the current character set, which makes the default comparison behavior case insensitive unless any or all of the operands are binary strings.

Example

```
SELECT INITCAP('====');

+----------------------+
| INITCAP('====') |
+----------------------+
| ====                 |
+----------------------+
1 row in set (Elapsed: 00:00:00.00)

SELECT INITCAP(123.1);
+--------------------+
| INITCAP(123.1) |
+--------------------+
| 123.1              |
+--------------------+
1 row in set (Elapsed: 00:00:00.00)

SELECT INITCAP('aa');
+------------------+
| INITCAP('aa') |
+------------------+
| Aa               |
```

```
+------------------+
1 row in set (Elapsed: 00:00:00.00)

SELECT INITCAP('a_a');
+--------------------+
| INITCAP('a_a') |
+--------------------+
| A_ A                    |
+--------------------+
1 row in set (Elapsed: 00:00:00.00)

SELECT INITCAP('aaa=');
+--------------------+
| INITCAP('aaa=') |
+--------------------+
| Aaa=                    |
+--------------------+
1 row in set (Elapsed: 00:00:00.00)

SELECT INITCAP('AAA');

+----------------------+
| INITCAP('AAA') |
+----------------------+
| Aaa                       |
+----------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.3.54 Regular expression function

5.1.5.3.54.1 regexp_ replace

grammar

```
regexp_
replace(source_char,pattern[,replace_string[,position[,occurrence[matc
h_option]]]])
```

explain

Use replace_ The string specified by string replaces the string in the source string that matches the regular expression specified by pattern.

function

Replace the string obtained by matching with the specified string.

Parameter Description

Table 515 Parameter Description

parameter	explain
source_ char	Source string. The data type supported by this parameter is consistent with the src parameter of the replace function in 8a.
pattern	Regular expressions. Each regular expression can contain up to 512 bytes. For specific grammar rules, please refer to the grammar rule explanation in version PCRE-7.8
replace_ string	Replace string. The replacement string can contain a numeric expression with a reverse reference (n, n has a value range of [1,9])
position	The starting position for matching, if not specified, defaults to 1, i.e. from source_ The first character of char begins to match. Position is a positive integer.
occurrence	The ordinal of regular matching. It is a non negative integer with a default value of 0. ● If specified as 0, replace all matching strings; ● If specified as an integer n, replace the string that was matched for the nth time;
match_ parameter	The default matching function behavior can be changed by setting this parameter. By default, '.' does not match line breaks, and the source string is treated as a single line. The parameter options are as follows: ● i: Case insensitive; ● c: Case sensitive; ● n: The period (.) does not match the line break symbol; ● m: Multi line mode;

parameter	explain
	● x: Extended mode, ignoring white space characters in regular expressions.

be careful

- When users specify multiple mutually exclusive parameter options at the same time, the system processes them according to the last parameter. User specified match_ When there are options other than parameter options (i, c, n, m, x), the system reports an error.
- regexp_ Replace function's replace_ string、position、occurrence、match_ The parameter parameters can all be omitted. If any of the above four parameters are omitted, all parameters after the omitted parameters cannot be set. If subsequent parameters need to be set, the value of the previous parameter of the set parameter must be given.
- Due to the use of backslashes ('') as escape characters in both SQL syntax and PCRE regular syntax. So in the pattern, two consecutive backslashes (' ') should be used as regular escape characters.
- regexp_ The replace function does not support recursion.
- If source_ If the char parameter is the result of the operation rather than the entity column of the table, and the operation result is greater than 512 bytes, the function will report an error.

For example:

```
CREATE VIEW v_ tb1 as SELECT * FROM tb1 WHERE v1 = (select
            regexp_ replace('000*111*999','\\*','-') from tb2 where
            v2=regexp_ replace(c1,'\\*','-'));
             SELECT * from alltype A left join (SELECT regexp_replace(v1, 'a',
    '')
             as c41 from v_ tb1)   B on A.V3 =
 regexp_ replace(B.c41,'\\*','-') ;
```

Example

Example 1: Using functions in a simple select query.

```
gbase> drop table if exists tb1;
Query OK, 0 rows affected
```

```
gbase> drop table if exists tb2;
Query OK, 0 rows affected

gbase> create table tb1(v1 varchar(25),c1 char(11),i1 int);
Query OK, 0 rows affected

gbase> insert into tb1 values('000-111-999','000*111*999',12);
Query OK, 1 row affected

gbase> create table tb2(v2 varchar(25),c2 char(11),i2 int);
Query OK, 0 rows affected

gbase> insert into tb2 values('000-111-999','000*111*999',12);
Query OK, 1 row affected

gbase> SELECT regexp_ replace('000-111-999','-','') from tb1;
+---------------------------------------+
| regexp_ replace('000-111-999','-','') |
+---------------------------------------+
| 000111999                             |
+---------------------------------------+
1 row in set
```

5.1.5.3.54.2 regexp_ like

grammar

```
REGEXP_ LIKE(source_char, pattern [, match_parameter])
```

function

Fuzzy match the specified string.

Return value

When the source string matches the regular expression specified in the pattern, the function returns 1, otherwise it returns 0.

parameter

Table 516 Parameter Description

parameter	explain

parameter	explain
source_ char	Source string. The data type supported by this parameter is consistent with the str parameter of the replace function in GBase 8a MPP Cluster.
pattern	Regular expressions. Only strings are supported, and each regular expression can contain up to 512 bytes.
match_ parameter	The default matching function behavior can be changed by setting this parameter. It can be a column name, and the content of the column cannot exceed the value range of the parameter. When used, this parameter is enclosed in single quotation marks, such as' i '. When the default and set to NULL do not match the newline character, the source string is treated as a single line. The parameter options are as follows: ● i: Case insensitive; ● c: Case sensitive; ● n: The period (.) matches the line break symbol; ● m: Multi line mode; ● x: Extended mode, ignoring white space characters in regular expressions. When a user specifies multiple mutually exclusive parameters (i, c) as optional items at the same time, the system processes them according to the last parameter that appears.

Use constraints

regexp_ Match of like function_ The parameter parameter can be omitted, and other parameters cannot be omitted.

5.1.5.3.54.3 regexp_ instr

grammar

```
REGEXP_
INSTR(source_char,pattern[,position[,occurrence[,return_opt[,match_parameter[,subexpr]]]]])
```

function

Obtain the position of the matching string.

purpose

Returns the position of the string in the source string that matches the regular expression specified by the pattern.

Parameter Explanation

Table -517 Parameter Description

parameter	explain
source_ char	Source string. The data type supported by this parameter is consistent with the str parameter of the replace function in GBase 8a MPP Cluster.
pattern	Regular expressions. Only strings are supported, and each regular expression can contain up to 512 bytes.
position	Start matching at. The default value is 1, that is, from the source_ The first character of char begins to match.
occurrence	The ordinal of regular matching. It is a positive integer, which can be a column name. It supports data types of integer and strings that can be converted to numbers. The conversion rules are consistent with the pos parameter conversion rules of the GBase 8a MPP Cluster's insert function. Decimals are not supported. If set to decimals, the rounding rule will apply. The default value is 1. ● If specified as 1, replace the first matching string; ● If specified as an integer n, replace the string that was matched for the nth time.
return_ opt	The type of the specified return value is a non negative integer, which can be a column name. The supported data types are integers and strings that can be converted to numbers. The conversion rules are consistent with the pos parameter conversion rules of the GBase 8a MPP Cluster system's insert function. Decimals are not supported. If set to decimals, the rounding rule will apply. The default value is 0. ● Specify as 0, and the return value is the position of the first character in the matching position. ● Specify n to return the position of the first character immediately following the matching string.
match_ parameter	The default matching function behavior can be changed by setting this parameter. It can be a column name, and the content of the column cannot exceed the value range of the parameter. When used, this parameter is enclosed in single quotation marks, such as' i '. When the default and set to NULL do not match the newline

parameter	explain
	character, the source string is treated as a single line. The parameter options are as follows: ● i: Case insensitive; ● c: Case sensitive; ● n: The period (.) matches the line break symbol; ● m: Multi line mode; ● x: Extended mode, ignoring white space characters in regular expressions. When a user specifies multiple mutually exclusive parameters (i, c) as optional items at the same time, the system processes them according to the last parameter that appears.
subexpr	For regular expressions containing subexpressions, it indicates which substring in the regular expression is the function target. Subexpr is a string fragment enclosed in parentheses in regular expressions, and subexpressions can be nested. Subexpressions are numbered in the order in which they appear in the left parenthesis. The range of values for this parameter is 0-9. If it exceeds 9, the function returns 0. It can be set to a string that can be converted into numbers, and the conversion rules are consistent with the pos parameter conversion rules of the GBase 8a MPP Cluster's insert function. Column names are not supported. Decimals are not supported. If set to decimals, the rounding rule will apply. The default is 0. ● If specified as 0, returns the position of the character that matches the regular expression, returns 1 for all matches, and returns 0 for no matches; ● If specified as greater than 0, returns the position of the specified substring. When the value is greater than the number of substrings, 0 is returned; ● If specified as null, the function returns null; ● When there are parentheses in the source string, follow the escape processing supported by the rule.

Use constraints

REGEXP_ The position, occurrence, and return of the INSTR function_ opt、matc h_ Both parameter and subexpr parameters can be omitted. If any of the above 5 parameters are omitted, all parameters after the omitted parameters cannot be set. If subsequent parameters need to be set, the value of the previous parameter of the set parameter must be provided.

5.1.5.3.54.4 regexp_ substr

grammar

```
REGEXP_ SUBSTR(source_char,pattern[,position[,occurrence[,match
_option[,subexpr]]])
```

function

Extract the substring of the specified string.

purpose

Find the string in the source string that matches the regular expression specified by the pattern.

parameter

Table 518 Parameter Description

parameter	explain
source_ char	Source string. The data type supported by this parameter is consistent with the str parameter of the replace function in GBase 8a MPP Cluster.
pattern	Regular expressions. Only strings are supported, and each regular expression can contain up to 512 bytes.
position	Start matching at. The default value is 1, that is, from the source_ The first character of char begins to match.
occurrence	The ordinal of regular matching. It is a positive integer, which can be a column name. It supports data types of integer and strings that can be converted to numbers. The conversion rules are consistent with the pos parameter conversion rules of the GBase 8a MPP Cluster's insert function. Decimals are not supported. If set to decimals, the rounding rule will apply. The default value is 1.

parameter	explain
	● If specified as 1, replace the first occurrence matched; ● If specified as an integer n, replace the occurrence of the nth match.
match_ parameter	The default matching function behavior can be changed by setting this parameter. It can be a column name, and the content of the column cannot exceed the value range of the parameter. When used, this parameter is enclosed in single quotation marks, such as' i '. When the default and set to null do not match the newline character, the source string is treated as a single line. The parameter options are as follows: ● i: Case insensitive; ● c: Case sensitive; ● n: The period (.) matches the line break symbol; ● m: Multi line mode; ● x: Extended mode, ignoring white space characters in regular expressions. When users specify multiple mutually exclusive parameters (i, c) as optional items, the system processes them according to the last parameter that appears.
subexpr	For regular expressions containing subexpressions, it indicates which substring in the regular expression is the function target. Subexpr is a string fragment enclosed in parentheses in regular expressions, and subexpressions can be nested. Subexpressions are numbered in the order in which they appear in the left parenthesis. The range of values for this parameter is 0-9. If it exceeds 9, the function returns 0. It can be set to a string that can be converted into numbers, and the conversion rules are consistent with the pos parameter conversion rules of the GBase 8a MPP Cluster's insert function. Column names are not supported. Decimals are not supported. If set to decimals, the rounding rule will apply. The default is 0. ● If specified as 0, returns the position of the string that matches the regular expression, returns 1 for all matches, and returns 0 for no matches; ● If specified as greater than 0, returns the position of the specified substring. When the value is greater than the number of substrings, 0 is returned; ● If specified as null, the function returns null; ● When there are parentheses in the source string, follow the

parameter	explain
	escape processing supported by the rule.

Use constraints

REGEXP_ The position, occurrence, and match of the SUBSTR function_ Both parameter and subexpr parameters can be omitted. If any of the above four parameters are omitted, all parameters after the omitted parameters cannot be set. If subsequent parameters need to be set, the value of the previous parameter of the set parameter must be provided.

5.1.5.4 Numerical function

5.1.5.4.1 arithmetic operator

Common arithmetic operators are available.

be careful

- If both parameters are integers, "-", "+", and "*" are evaluated with BIGINT (64 bit) precision and return the result.
- If one parameter is an unsigned integer and the other parameters are integers, the result is an unsigned integer.

5.1.5.4.2 +Addition

Example

Example 1: Both operands are integers.

```
gbase> SELECT 3+5 FROM dual;
+-----+
| 3+5 |
+-----+
|   8 |
+-----+
1 row in set
```

5.1.5.4.3 -Subtraction

Example

Example 1: Both operands are integers.

```
gbase> SELECT 3-5 FROM dual;
+-----+
| 3-5 |
+-----+
|  -2 |
+-----+
1 row in set
```

5.1.5.4.4 -One yuan reduction

Function Description

Change the symbol of the parameter.

Example

Example 1: The operand is an integer.

```
gbase> SELECT - 2 FROM dual;
+-----+
| - 2 |
+-----+
|  -2 |
+-----+
1 row in set
```

be careful

If the operand is of type BIGINT, then the return value is also of type BIGINT.

5.1.5.4.5 *Multiplication

Example

Example 1: Both operands are integers.

```
gbase> SELECT 3*5 FROM dual;
+-----+
| 3*5 |
+-----+
|  15 |
+-----+
1 row in set
```

5.1.5.4.6 /Division

Example

Example 1: Both operands are integers.

```
gbase> SELECT 3/5 FROM dual;
+--------+
| 3/5    |
+--------+
| 0.6000 |
+--------+
1 row in set
```

Example 2: Divided by 0, the return value is NULL.

```
gbase> SELECT 102/(1-1) FROM dual;
+-----------+
| 102/(1-1) |
+-----------+
|      NULL |
+-----------+
1 row in set
```

 **explain**

Division will only perform arithmetic calculations using BIGINT when executed in the context of a result being converted to an integer.

5.1.5.4.7 DIV integer division

Example

Example 1: Both operands are integers.

```
gbase> SELECT 5 DIV 2 FROM dual;
+---------+
| 5 DIV 2 |
+---------+
|       2 |
+---------+
1 row in set
```

5.1.5.4.8 Mathematical function

- All mathematical functions return NULL in the event of an error.
- The degree of support of mathematical functions for decimal type operations is

explained as follows:

The following mathematical function operations support decimal precise type operations, with a minimum precision of 16 bits when the returned result is of decimal type. This function is determined by the parameter gbase_ decimal_ Calculation control, default to 0 off, function operation result is of type double; When the value is set to 1, it is enabled. The return rules are shown in the table below:

Mathematical function	X type	Type y	Return Type
Exp(x)	Int/decimal	-	decimal
Sqrt(x) Ln(x) Log(x)/log2(x)/log10(x)	double	-	double
Log(x,y)	Int/decimal	Int/decimal	decimal
Pow(x,y)	Int/decimal	double	double
	double	Int/decimal/double	double

Note:

1. The default precision of the return value type is 16 bits.
2. When the maximum precision of the input parameter is less than 16 bits, the return value type precision is 16 bits.
3. When the maximum precision of the input parameter is greater than 16 bits, the return value type precision of the function is the maximum precision of the input parameter.
4. When the result is of type decimal, the range of the result becomes smaller, indicating the maximum representation range of decimal (65, precision). The higher the accuracy, the smaller the range.
5. gbase_ decimal_ When the calculation parameter is turned on and the result type is decimal, it is not compatible with the pre upgrade version of the result set.

5.1.5.4.8.1 ABS(X)

Function Description

Returns the absolute value of X. This function supports the use of BIGINT values.

Example

Example 1: X is a positive number.

```
gbase> SELECT ABS(2) FROM dual;
+--------+
| ABS(2) |
+--------+
|      2 |
+--------+
1 row in set
```

Example 2: X is a negative number.

```
gbase> SELECT ABS(-32) FROM dual;
+----------+
| ABS(-32) |
+----------+
|       32 |
+----------+
1 row in set
```

5.1.5.4.8.2 ACOS(X)

Function Description

Returns the arccosine of X, that is, a value with a cosine value of X.

If X is not within the range of -1 to 1, return NULL.

Example

Example 1: X is a positive number.

```
gbase> SELECT ACOS(1) FROM dual;
+---------+
| ACOS(1) |
+---------+
|       0 |
+---------+
1 row in set
```

Example 2: X is greater than 1.

```
gbase> SELECT ACOS(1.0001) FROM dual;
+--------------+
| ACOS(1.0001) |
+--------------+
|         NULL |
+--------------+
1 row in set
```

Example 3: X is 0.

```
gbase> SELECT ACOS(0) FROM dual;
+-----------------+
| ACOS(0)         |
+-----------------+
| 1.5707963267949 |
+-----------------+
1 row in set
```

5.1.5.4.8.3 ASIN(X)

Function Description

Returns the arcsine of X, that is, a value with a sine value of X.

If X is not within the range of -1 to 1, return NULL.

Example

Example 1: X is a decimal.

```
gbase> SELECT ASIN(0.2) FROM dual;
+-------------------+
| ASIN(0.2)         |
+-------------------+
| 0.201357920790331 |
+-------------------+
1 row in set
```

Example 2: X is greater than 1.

```
gbase> SELECT ASIN(2) FROM dual;
+---------+
| ASIN(2) |
+---------+
|    NULL |
+---------+
1 row in set
```

5.1.5.4.8.4 ATAN(X)

Function Description

Returns the arctangent of X, that is, the value whose tangent value is X.

Example

Example 1: X is a positive integer.

```
gbase> SELECT ATAN(2) FROM dual;
+------------------+
| ATAN(2)          |
+------------------+
| 1.10714871779409 |
+------------------+
1 row in set
```

Example 2: X is a negative integer.

```
gbase> SELECT ATAN(-2) FROM dual;
+------------------+
| ATAN(-2)         |
+------------------+
| -1.10714871779409 |
+------------------+
1 row in set
```

5.1.5.4.8.5 ATAN(Y,X), ATAN2(Y,X)

Function Description

Returns the arctangent of two variables X and Y. It is similar to calculating the arctangent of Y/X, where the symbols of the two parameters are used to determine the quadrant in which the result is located.

Example

Example 1: Returns the arctangent of "-2/2".

```
gbase> SELECT ATAN(-2,2) FROM dual;
+--------------------+
| ATAN(-2,2)         |
+--------------------+
| -0.785398163397448 |
+--------------------+
1 row in set
```

Example 2: Returns the arctangent of "PI()/0".

```
gbase> SELECT ATAN2(PI(),0) FROM dual;
+-----------------+
| ATAN2(PI(),0)   |
+-----------------+
| 1.5707963267949 |
+-----------------+
1 row in set
```

5.1.5.4.8.6 CEILING(X), CEIL(X)

Function Description

Returns the smallest integer not less than X.

Example

Example 1: X is a positive number.

```
gbase> SELECT CEILING(1.23) FROM dual;
+---------------+
| CEILING(1.23) |
+---------------+
|             2 |
+---------------+
1 row in set
```

Example 2: X is a negative number.

```
gbase> SELECT CEIL(-1.23) FROM dual;
+-------------+
| CEIL(-1.23) |
+-------------+
|          -1 |
+-------------+
1 row in set
```

5.1.5.4.8.7 COS(X)

Function Description

Returns the cosine of X, where X is given in radians.

Example

Example 1: X is PI().

```
gbase> SELECT COS(PI()) FROM dual;
+-----------+
| COS(PI()) |
+-----------+
|        -1 |
+-----------+
1 row in set
```

5.1.5.4.8.8 COT(X)

Function Description

Returns the cotangent of X.

Example

Example 1: X is a positive number.

```
gbase> SELECT COT(12) FROM dual;
+--------------------+
| COT(12)            |
+--------------------+
| -1.57267340639769 |
+--------------------+
1 row in set
```

Example 2: X is 0.

```
gbase> SELECT COT(0) FROM dual;
+--------+
| COT(0) |
+--------+
|   NULL |
+--------+
1 row in set
```

5.1.5.4.8.9 CRC32(expr)

Function Description

Calculate the cyclic redundancy code check value and return a 32-bit unsigned value.

Expr should be a string and will be processed as a string if it is not a string (if successfully converted to a string type).

If the parameter is NULL, the result is NULL.

Example

Example 1: expr is a string.

```
gbase> SELECT CRC32('GBase') FROM dual;
+----------------+
| CRC32('GBase') |
+----------------+
|     3594920295 |
+----------------+
1 row in set
```

Example 2: expr is a number.

```
gbase> SELECT CRC32(1.034) FROM dual;
+--------------+
| CRC32(1.034) |
+--------------+
|   1481567290 |
+--------------+
1 row in set
```

5.1.5.4.8.10 DEGREES(X)

Function Description

Convert parameter X from radians to angles.

Example

Example 1: X is PI().

```
gbase> SELECT DEGREES(PI())FROM dual;
+---------------+
| DEGREES(PI()) |
+---------------+
|           180 |
+---------------+
1 row in set
```

5.1.5.4.8.11 EXP(X)

Function Description

Returns a value with a cardinality of e and a power of X, that is, returns e to the power of X.

Example

Example 1: Returns e to the 2nd power.

```
gbase> SELECT EXP(2) FROM dual;
+------------------+
| EXP(2)           |
+------------------+
| 7.38905609893065 |
+------------------+
1 row in set
```

Example 2: Returns e to the power of -2.

```
gbase> SELECT EXP(-2) FROM dual;
+--------------------+
| EXP(-2)            |
+--------------------+
| 0.135335283236613 |
+--------------------+
1 row in set
```

5.1.5.4.8.12 FLOOR(X)

Function Description

Returns the maximum integer value not greater than X.

If the parameter X is NULL, the return result is NULL.

Example

Example 1: X is a positive number.

```
gbase> SELECT FLOOR(1.23) FROM dual;
+-------------+
| FLOOR(1.23) |
+-------------+
|           1 |
+-------------+
1 row in set
```

Example 2: X is a negative number.

```
gbase> SELECT FLOOR(-1.23) FROM dual;
+--------------+
| FLOOR(-1.23) |
+--------------+
|           -2 |
+--------------+
1 row in set
```

Example 3: X is NULL.

```
gbase> SELECT FLOOR(NULL) FROM dual;
+-------------+
| FLOOR(NULL) |
+-------------+
|        NULL |
+-------------+
1 row in set
```

5.1.5.4.8.13 LN(X)

Function Description

Returns the natural logarithm of X.

Example

Example 1: Return the natural logarithm of 2.

```
gbase> SELECT LN(2) FROM dual;
+-------------------+
| LN(2)             |
+-------------------+
| 0.693147180559945 |
+-------------------+
1 row in set
```

Example 2: Return the natural logarithm of - 2.

```
gbase> SELECT LN(-2) FROM dual;
+--------+
| LN(-2) |
+--------+
|   NULL |
+--------+
1 row in set
```

5.1.5.4.8.14 LOG(X), LOG(B,X)

Function Description

If called with a parameter, it returns the natural logarithm of X.

This function has the same meaning as LN (X).

Example

Example 1: Return the natural logarithm of 2.

```
gbase> SELECT LOG(2) FROM dual;
+-------------------+
| LOG(2)            |
+-------------------+
| 0.693147180559945 |
+-------------------+
1 row in set
```

Example 2: Return the natural logarithm of - 2.

```
gbase> SELECT LOG(-2) FROM dual;
+---------+
| LOG(-2) |
+---------+
|    NULL |
+---------+
1 row in set
```

Example 3: If called with two parameters, this function returns the logarithm of X based on B.

```
gbase> SELECT LOG(2,65536) FROM dual;
+--------------+
| LOG(2,65536) |
+--------------+
|           16 |
+--------------+
1 row in set
```

Example 4: LOG (B, X) is equivalent to LOG (X)/LOG (B).

```
gbase> SELECT LOG(1,100) FROM dual;
+------------+
| LOG(1,100) |
+------------+
|       NULL |
+------------+
1 row in set
```

5.1.5.4.8.15 LOG2(X)

Function Description

Returns the base 2 logarithm of X. Usually used to calculate how many bits of storage a number requires.

Example

Example 1: Returns the logarithm of "65536" based on 2.

```
gbase> SELECT LOG2(65536) FROM dual;
+-------------+
| LOG2(65536) |
+-------------+
|          16 |
+-------------+
1 row in set
```

Example 2: Returns the logarithm of "-100" based on 2.

```
gbase> SELECT LOG2(-100) FROM dual;
+------------+
| LOG2(-100) |
+------------+
|       NULL |
+------------+
1 row in set
```

5.1.5.4.8.16 LOG10(X)

Function Description

Returns the logarithm of X based on 10.

Example

Example 1: Returns the logarithm of '2' based on 10.

```
gbase> SELECT LOG10(2) FROM dual;
+-------------------+
| LOG10(2)          |
+-------------------+
| 0.301029995663981 |
+-------------------+
1 row in set
```

Example 2: Returns the logarithm of "100" with a base of 10.

```
gbase> SELECT LOG10(100) FROM dual;
+------------+
| LOG10(100) |
+------------+
|          2 |
+------------+
1 row in set
```

Example 3: Returns the logarithm of "-100" with a base of 10.

```
gbase> SELECT LOG10(-100) FROM dual;
+-------------+
| LOG10(-100) |
+-------------+
|        NULL |
+-------------+
1 row in set
```

5.1.5.4.8.17 MOD(N,M)，N % M，N MOD M

Function Description

Take the mold. Returns the remainder of N divided by M.

Example

Example 1: Returns the remainder of 234 divided by 10.

```
gbase> SELECT MOD(234, 10) FROM dual;
+--------------+
| MOD(234, 10) |
+--------------+
|            4 |
+--------------+
1 row in set
```

Example 2: Returns the remainder of 253 divided by 7.

```
gbase> SELECT 253 % 7 FROM dual;
+---------+
| 253 % 7 |
+---------+
|       1 |
+---------+
1 row in set
```

Example 3: MOD (29,9) and 29 MOD 9 have the same results.

```
gbase> SELECT MOD(29,9) FROM dual;
+-----------+
| MOD(29,9) |
+-----------+
|         2 |
+-----------+
1 row in set

gbase> SELECT 29 MOD 9 FROM dual;
+----------+
| 29 MOD 9 |
+----------+
|        2 |
+----------+
1 row in set
```

Example 4: MOD() also applies to the decimal part, returning the exact remainder after a division operation.

```
gbase> SELECT MOD(34.5,3) FROM dual;
+-------------+
| MOD(34.5,3) |
+-------------+
|         1.5 |
+-------------+
1 row in set
```

Example 5: Three forms of representation for mold taking.

```
gbase> SELECT 253 % 7, MOD(253,7), 253 MOD 7 FROM dual;
+---------+------------+-----------+
| 253 % 7 | MOD(253,7) | 253 MOD 7 |
+---------+------------+-----------+
|       1 |          1 |         1 |
+---------+------------+-----------+
1 row in set
```

5.1.5.4.8.18 PI()

Function Description

Returns the PI value (pi). The default display is 6 decimal places, but within GBase 8a MPP Cluster, PI uses all double precision.

Example

Example 1: Returns the value of PI.

```
gbase> SELECT PI() FROM dual;
+----------+
| PI()     |
+----------+
| 3.141593 |
+----------+
1 row in set
```

5.1.5.4.8.19 POW(X,Y), POWER(X,Y)

Function Description

Returns X to the Y-th power.

Example

Example 1: Returns 2 to the 2nd power.

```
gbase> SELECT POW(2,2) FROM dual;
+----------+
| POW(2,2) |
+----------+
|        4 |
+----------+
1 row in set
```

Example 2: Returns 2 to the power of -2.

```
gbase> SELECT POW(2,-2) FROM dual;
+-----------+
| POW(2,-2) |
+-----------+
|      0.25 |
+-----------+
1 row in set
```

5.1.5.4.8.20 RADIANS(X)

Function Description

Convert parameter X from angle to radian, and then return.

Example

Example 1: Returns the radian corresponding to 90 degrees.

```
gbase> SELECT RADIANS(90) FROM dual;
+-----------------+
| RADIANS(90)     |
+-----------------+
| 1.5707963267949 |
+-----------------+
1 row in set
```

5.1.5.4.8.21 RAND(), RAND(N)

Function Description

Returns a random floating point number in the range 0 to 1.0.

If an integer parameter N is specified, it is used as a seed value (used to generate a repeatable numerical value).

Example

Example 1: Return a random floating point number.

```
gbase> SELECT RAND() FROM dual;
+------------------+
| RAND()           |
+------------------+
| 0.926571502281885 |
+------------------+
1 row in set

gbase> SELECT RAND() FROM dual;
+------------------+
| RAND()           |
+------------------+
| 0.81284204853032 |
+------------------+
1 row in set

gbase> SELECT RAND() FROM dual;
+------------------+
| RAND()           |
+------------------+
| 0.323826807852673 |
+------------------+
1 row in set
```

Example 2: Return a random floating point number, run RAND (20) again, and the result is the same as the last time.

```
gbase> SELECT RAND(20) FROM dual;
+------------------+
| RAND(20)         |
+------------------+
| 0.158882612510475 |
+------------------+
1 row in set

gbase> SELECT RAND(20) FROM dual;
+------------------+
| RAND(20)         |
+------------------+
| 0.158882612510475 |
+------------------+
1 row in set
```

explain

In an ORDER BY clause, the RAND() value cannot be applied to a column, as ORDER BY will repeatedly calculate the column multiple times. Users can retrieve rows in any order.

5.1.5.4.8.22 ROUND(X), ROUND(X,D)

Function Description

ROUND (X) returns the value of parameter X rounded to the nearest integer.

The X value returned by ROUND (X, D) is rounded to the D-th place after the decimal point.

be careful

ROUND the double type and follow the "rounding to even" rule:

- When the rounded number is less than 5, the number is rounded off;
- When the rounded number is greater than 5, it is rounded;
- When the number being rounded off is equal to 5, it is important to look at the number before 5. If it is an odd number, it is rounded up, and if it is an even number, it is rounded off. This means that after rounding off, all the numbers at the end become even; If there is any number after 5 that is not "0", then regardless of whether the number before 5 is odd or even, it should be carried forward.

If the D value is negative, the retained X value is the D digit to the left of the decimal point.

Example

Example 1: X is "-1.23" and the return result is -1.

```
gbase> SELECT ROUND(-1.23) FROM dual;
+--------------+
| ROUND(-1.23) |
+--------------+
|           -1 |
+--------------+
1 row in set
```

Example 2: X is "-1.58" and the return result is -2.

```
gbase> SELECT ROUND(-1.58) FROM dual;
+--------------+
| ROUND(-1.58) |
+--------------+
|           -2 |
+--------------+
1 row in set
```

Example 3: X is "1.58" and the return result is 2.

```
gbase> SELECT ROUND(1.58) FROM dual;
+-------------+
| ROUND(1.58) |
+-------------+
|           2 |
+-------------+
1 row in set
```

Example 4: Round "1.298" to one decimal place.

```
gbase> SELECT ROUND(1.298, 1) FROM dual;
+-----------------+
| ROUND(1.298, 1) |
+-----------------+
|             1.3 |
+-----------------+
1 row in set
```

Example 5: Round "1.298" to 0 decimal places.

```
gbase> SELECT ROUND(1.298, 0) FROM dual;
+-----------------+
| ROUND(1.298, 0) |
+-----------------+
|               1 |
+-----------------+
1 row in set
```

Example 6: Round "23.298" to the nearest decimal place, leaving "-1" digits after the decimal point, which is the one digit number.

```
gbase> SELECT ROUND(23.298, -1) FROM dual;
+------------------+
| ROUND(23.298, -1) |
+------------------+
|                20 |
+------------------+
1 row in set
```

explain

- The return value type is the same as the type of the first parameter.
- When the first parameter is DECIMAL, ROUND() uses an exact calculation library for precise calculations.
- For precise numerical values, ROUND() uses the rules of "rounding" or "rounding to the nearest number".
- If the decimal part of a value is. 5 or greater, it is rounded up to the next integer (such as 1.5 rounded to 2). For negative numbers, it is rounded down to the next negative number (such as -1.5 rounded to -2).
- If the decimal part of a value is smaller than. 5, it is rounded down to the previous integer (such as 1.4 rounded to 1). For negative numbers, it is rounded up to the previous negative number (such as -1.4 rounded to -1).
- For approximate numbers, the results are determined based on the C library. In many systems, this means that the use of ROUND() follows the rule of rounding to the nearest even number.
- A value with any decimal part will be rounded to the nearest even integer. For 25E-1, it believes that 20E-1 is closest to it.

Example 7: Differences in rounding between precise and approximate values.

```
gbase> SELECT ROUND(2.5), ROUND(25E-1) FROM dual;
+------------+--------------+
| ROUND(2.5) | ROUND(25E-1) |
+------------+--------------+
|          3 |            2 |
+------------+--------------+
1 row in set
```

Example 8: For the rounding of DECIMAL columns and precise values, the principle of rounding up to half is used. When the decimal part of a numerical value is 0.5 or greater, it is rounded to the nearest integer.

```
gbase> SELECT ROUND(2.5), ROUND(-2.5) FROM dual;
+------------+-------------+
| ROUND(2.5) | ROUND(-2.5) |
+------------+-------------+
|          3 |          -3 |
+------------+-------------+
1 row in set
```

5.1.5.4.8.23 SIGN(X)

Function Description

Returns 1, 0, or -1 based on whether the X value is positive, 0, or negative.

Example

Example 1: X is a negative number and returns -1.

```
gbase> SELECT SIGN(-32) FROM dual;
+-----------+
| SIGN(-32) |
+-----------+
|        -1 |
+-----------+
1 row in set
```

Example 2: X is' 0 'and returns 0.

```
gbase> SELECT SIGN(0) FROM dual;
+---------+
| SIGN(0) |
+---------+
|       0 |
+---------+
1 row in set
```

Example 3: X is a positive number and returns 1.

```
gbase> SELECT SIGN(234) FROM dual;
+-----------+
| SIGN(234) |
+-----------+
|         1 |
+-----------+
1 row in set
```

5.1.5.4.8.24 SIN(X)

Function Description

Returns the sine of X, where X is given in radians.

Example

Example 1: Returns the sine of 'PI()'.

```
gbase> SELECT SIN(PI()) FROM dual;
+----------------------+
| SIN(PI())            |
+----------------------+
| 1.22464679914735e-16 |
+----------------------+
1 row in set
```

5.1.5.4.8.25 SQRT(X)

Function Description

Returns the non negative square root of X.

Example

Example 1: Returns the square root of '4'.

```
gbase> SELECT SQRT(4) FROM dual;
+---------+
| SQRT(4) |
+---------+
|       2 |
+---------+
1 row in set
```

Example 2: Returns the square root of "20".

```
gbase> SELECT SQRT(20) FROM dual;
+------------------+
| SQRT(20)         |
+------------------+
| 4.47213595499958 |
+------------------+
1 row in set
```

5.1.5.4.8.26 TAN(X)

Function Description

Returns the tangent of X, where X is given in radians.

Example

Example 1: Returns the tangent value of "PI()+1".

```
gbase> SELECT TAN(PI()+1) FROM dual;
+-----------------+
| TAN(PI()+1)     |
+-----------------+
| 1.5574077246549 |
+-----------------+
1 row in set
```

5.1.5.4.8.27 TRUNCATE(X,D)

Function Description

Returns the number X truncated to D decimal places. D is an optional parameter, with a default value of 0.

If D is 0, the result will not include the decimal point and decimal part.

If D is a negative number, it means truncating (zeroing) all low values after the D-th digit to the left of the decimal point of the X value.

Example

Example 1: X is "1.223", with one decimal place reserved.

```
gbase> SELECT TRUNCATE(1.223,1) FROM dual;
+------------------+
| TRUNCATE(1.223,1) |
+------------------+
|              1.2 |
+------------------+
1 row in set
```

Example 2: X is "1.999", with one decimal place reserved.

```
gbase> SELECT TRUNCATE(1.999,1) FROM dual;
+------------------+
| TRUNCATE(1.999,1) |
+------------------+
|              1.9 |
+------------------+
1 row in set
```

Example 3: D is' 0 ', and the return value does not contain the decimal point or part.

```
gbase> SELECT TRUNCATE(1.999,0) FROM dual;
+------------------+
| TRUNCATE(1.999,0) |
+------------------+
|                1 |
+------------------+
1 row in set
```

Example 4: X is "-1.999", with one decimal place reserved.

```
gbase> SELECT TRUNCATE(-1.999,1) FROM dual;
+-------------------+
| TRUNCATE(-1.999,1) |
+-------------------+
|              -1.9 |
+-------------------+
1 row in set
```

Example 5: D is -2, with tens of bits zeroed.

```
gbase> SELECT TRUNCATE(122,-2) FROM dual;
+------------------+
| TRUNCATE(122,-2) |
+------------------+
|                100 |
+------------------+
1 row in set
```

Example 6: X is an expression.

```
gbase> SELECT TRUNCATE(10.28*100,0) FROM dual;
+-----------------------+
| TRUNCATE(10.28*100,0) |
+-----------------------+
|                  1028 |
+-----------------------+
1 row in set
```

5.1.5.5 Date and Time Functions

summary

This section describes the functions that can be used to manipulate time values. Reference date and time types to obtain the range of values that each date and time type can express in a valid format.

The function that returns the current date or time is equal to the value at the beginning of query execution, executed only once. This means that multiple references to a function like NOW () in a single query will always yield the same result. This principle also applies to CURDATE(), CURTIME(), UTC_ DATE()、UTC_ TIME()、UTC_ TIMESTAMP() and their synonyms.

CURRENT_ TIMESTAMP()、CURRENT_ TIME()、CURRENT_ DATE() and From_ UNIXTIME() returns the current time zone, which is the same as time_ The zone system variable is the same. And UNIX_ TIMESTAMP() assumes that its parameter is the datetime value of the current time zone.

Example

Example 1: Returns the current date and time.

```
gbase> SELECT NOW() FROM dual;
+---------------------+
| NOW()               |
+---------------------+
```

```
| 2020-04-01 14:53:55 |
+---------------------+
1 row in set
```

Example 2: Returns the current date.

```
gbase> SELECT CURDATE() FROM dual;
+------------+
| CURDATE()  |
+------------+
| 2020-04-01 |
+------------+
1 row in set
```

Example 3: Returns the current time.

```
gbase> SELECT CURTIME() FROM dual;
+-----------+
| CURTIME() |
+-----------+
| 15:37:04  |
+-----------+
1 row in set
```

Example 4: Returns the current UTC date.

```
gbase> SELECT UTC_ DATE() FROM dual;
+------------+
| UTC_ DATE() |
+------------+
| 2020-04-01 |
+------------+
1 row in set
```

Example 5: Returns the current UTC time.

```
gbase> SELECT UTC_ TIME() FROM dual;
+------------+
| UTC_ TIME() |
+------------+
| 07:37:32   |
+------------+
1 row in set
```

Example 6: Returns the current UTC timestamp (date+time).

```
gbase> SELECT UTC_ TIMESTAMP() FROM dual;
+---------------------+
| UTC_ TIMESTAMP()    |
```

```
+---------------------+
| 2020-04-01 07:37:59 |
+---------------------+
1 row in set
```

Example 7: Returns the current timestamp (date+time).

```
gbase> SELECT CURRENT_ TIMESTAMP() FROM dual;
+---------------------+
| CURRENT_ TIMESTAMP() |
+---------------------+
| 2020-04-01 15:38:14 |
+---------------------+
1 row in set
```

Example 8: Returns the current time.

```
gbase> SELECT CURRENT_ TIME() FROM dual;
+----------------+
| CURRENT_ TIME() |
+----------------+
| 15:38:26       |
+----------------+
1 row in set
```

Example 9: Returns the current date.

```
gbase> SELECT CURRENT_ DATE() FROM dual;
+----------------+
| CURRENT_ DATE() |
+----------------+
| 2020-04-01     |
+----------------+
1 row in set
```

Example 10: Obtain multiple current date and time values at once.

```
gbase> SELECT NOW(),CURRENT_ DATE() as cur_
d,CURTIME(),CURRENT_ TIMESTAMP() as cur_ ts FROM dual;
+---------------------+------------+-----------+---------------------+
| NOW()               | cur_ d     | CURTIME() | cur_ ts             |
+---------------------+------------+-----------+---------------------+
| 2020-04-01 17:42:43 | 2020-04-01 | 17:42:43  | 2020-04-01 17:42:43 |
+---------------------+------------+-----------+---------------------+
1 row in set
```

5.1.5.5.1 Daylight Saving Time

summary

Summer time refers to setting the clock one hour earlier in the summer when the sun rises to improve the use of sunlight. Supporting time zones is a prerequisite for supporting daylight saving time, and different time zones have different support for daylight saving time. Daylight saving time can only take effect within the time zone in which it is used.

explain

- system_ time_ Zone: Display the time zone of the operating system.
- time_ Zone: The time zone used by GBase in the current session, a session level variable, which can be set using default time zone in the configuration file or set time_ Make modifications to the zone. time_ The value of zone has three forms:
- System： time_ The value of zone is the same as that of system_ time_ The value of zone is the same. If it is not set in the configuration file, this value is the default value.
- UTC: represents the offset, with a range of [-12:59, 13:00], for example, '+8:00' represents the East Eighth District.
- time_ zone_ Name: Time zone name, from time_ zone_ The name of the time zone that can be found in the name, for example, Asia/Shanghai represents the time zone where Shanghai is located.

Example

Example 1: Using different time zones.

```
gbase> set time_ zone='+8:00';
Query OK, 0 rows affected (Elapsed: 00:00:00.00)

gbase> show variables like '%time_ zone%';
+------------------+--------+
| Variable_ name       | Value    |
+------------------+--------+
| system_ time_ zone | CST       |
| time_ zone              | +08:00 |
+------------------+--------+
2 rows in set (Elapsed: 00:00:00.00)
```

```
gbase> set time_ zone='SYSTEM';
```

```
Query OK, 0 rows affected (Elapsed: 00:00:00.00)

gbase> show variables like '%time_ zone%';
+------------------+--------+
| Variable_ name       | Value    |
+------------------+--------+
| system_ time_ zone | CST       |
| time_ zone               | SYSTEM |
+------------------+--------+
2 rows in set (Elapsed: 00:00:00.00)
```

```
gbase> set time_ zone='US/Central';
Query OK, 0 rows affected (Elapsed: 00:00:00.00)

gbase> show variables like '%time_ zone%';
+------------------+------------+
| Variable_ name       | Value           |
+------------------+------------+
| system_ time_ zone | CST               |
| time_ zone               | US/Central |
+------------------+------------+
2 rows in set (Elapsed: 00:00:00.00)
```

Example 2: Modifying the system time zone for comparison.

```
cp    /usr/share/zoneinfo/US/Pacific    /etc/localtime

gbase> set time_ zone='SYSTEM';
Query OK, 0 rows affected (Elapsed: 00:00:00.00)

gbase> show variables like '%zone%';
+------------------+--------+
| Variable_ name       | Value    |
+------------------+--------+
| system_ time_ zone | CST       |
| time_ zone               | SYSTEM |
+------------------+--------+
2 rows in set (Elapsed: 00:00:00.00)

gbase> \! date
Monday, June 7th, 2021 14:02:59 CST
gbase> SELECT now() FROM dual;
+---------------------+
| now()                          |
+---------------------+
```

```
| 2021-06-07 14:03:01 |
+---------------------+
1 row in set (Elapsed: 00:00:00.04)

gbase> \! date -u +%s
one billion five hundred and eighty-five million seven hundred and nineteen
thousand two hundred and forty-nine
gbase> SELECT from_ unixtime(1585719249) FROM dual;
+--------------------------+
| from_ unixtime(1585719249) |
+--------------------------+
| 2020-04-01 13:34:09          |
+--------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> \! date
Monday, June 7th, 2021 14:02:59 CST
```

Example 3: During daylight saving time, the time in gbase is consistent with the system time.

```
//Modify the system time zone
cp    /usr/share/zoneinfo/US/Pacific    /etc/localtime
//Change the system time to within daylight saving time:
# date -s '2020-3-11 5:00:00'
Wed Mar 11 05:00:00 PST 2020
# hwclock -w
# date    '+%Y-%m-%d %H:%M:%S'
2020-03-11 05:04:22

gbase> set time_ zone='SYSTEM';
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> show variables like '%zone%';
+------------------+--------+
| Variable_ name       | Value    |
+------------------+--------+
| system_ time_ zone | CST       |
| time_ zone              | SYSTEM |
+------------------+--------+
2 rows in set (Elapsed: 00:00:00.00)
gbase> \! date '+%Y-%m-%d %H:%M:%S'
2020-03-11 05:06:06
gbase> SELECT now() FROM dual;
+--------------------+
| now()                        |
```

```
+---------------------+
| 2020-03-11 05:06:10 |
+---------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> \! date -u +%s
one billion five hundred and eighty-three million eight hundred and seventy-four
thousand four hundred and twenty-three
gbase> SELECT from_ unixtime(1583874423) FROM dual;
+---------------------------+
| from_ unixtime(1583874423) |
+---------------------------+
| 2020-03-11 05:07:03       |
+---------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.5.2 ADDDATE()

grammar

ADDDATE(date,INTERVAL expr type), ADDDATE(expr,days)

Function Description

The keywords INTERVAL and type classifiers are not case sensitive.

When calling the second parameter INTERVAL, ADDDATe() is equivalent to DATE_ ADD()。

In ADDDATE (expr, days), expr is a date or datetime expression, and days is the number of days to add the date to expr. By default, increase the number of days.

Example

Example 1: ADDDATE (date, INTERVAL expr type), expr is the date, returns the date after 31 days of increase.

```
gbase> SELECT ADDDATE('2020-01-02', INTERVAL 31 DAY) FROM
dual;
+----------------------------------------+
| ADDDATE('2020-01-02', INTERVAL 31 DAY) |
+----------------------------------------+
| 2020-02-02 00:00:00                    |
+----------------------------------------+
1 row in set
```

Example 2: ADDDATE (expr, days), returns a date that has been added for 31 days.

```
gbase> SELECT ADDDATE('2020-01-02', 31 ) FROM dual;
+-----------------------------+
| ADDDATE('2020-01-02', 31 ) |
+-----------------------------+
| 2020-02-02 00:00:00 |
+-----------------------------+
1 row in set
```

5.1.5.5.3 ADDTIME(expr,expr2)

Function Description

Add expr2 to expr and return the result.

Expr is a time or datetime expression, and expr2 is a time expression.

Example

Example 1: expr is the date time.

```
gbase> SELECT ADDTIME('2020-01-02 23:59:59.999999','1 1:1:1.000002')
FROM dual;
+-----------------------------------------------------------+
| ADDTIME('2020-01-02 23:59:59.999999','1 1:1:1.000002') |
+-----------------------------------------------------------+
| 2020-01-04 01:01:01.000001 |
+-----------------------------------------------------------+
 1 row in set
```

5.1.5.5.4 ADD_ MONTHS(date,number[,mode])

Function Description

ADD_ The MONTHS (date, number [, mode]) function adds the specified number of months to a date, where the days in the date remain unchanged.

If the start date is the last day of a month and the number of days in the result month is less than the number of days in the start month, the result will return the last day of the corresponding result month.

Table 519 Parameter Description

Parameter Name	Description
date	Descripti is a date value.

Parameter Name	Description
number	The number of months added is negative if it is the number of months to subtract.
mode	If the start date is the last day of a month and the number of days in the result month is greater than the number of days in the start month, the result is returned based on the mode value. If mode is 0, return the day corresponding to the start date; If it is 1, return the last day of the corresponding result month. The default value is 0.

Example

Example 1: Add 3 months to the current date and time, and the days in the date remain unchanged.

```
gbase> SELECT NOW(),ADD_ MONTHS(NOW(),3) AS f_ Show FROM
dual;
+---------------------+---------------------+
| NOW()               | f_ Show             |
+---------------------+---------------------+
| 2020-04-01 13:55:56 | 2020-07-01 13:55:56 |
+---------------------+---------------------+
1 row in set
```

Example 2: When the number of months added is negative, it is equivalent to the number of months in advance.

```
gbase> SELECT NOW(),ADD_ MONTHS(NOW(),-3) AS f_ Show FROM
dual;
+---------------------+---------------------+
| NOW()               | f_ Show             |
+---------------------+---------------------+
| 2020-04-01 13:56:26 | 2020-01-01 13:56:26 |
+---------------------+---------------------+
1 row in set
```

Example 3: By using to_ The date function is converted to a date type, followed by the specified month.

```
gbase> SELECT ADD_
MONTHS(TO_DATE('2020-3-15','YYYY-MM-DD'),3) AS f_ Show FROM
dual;
+------------+
| f_ Show    |
+------------+
| 2020-06-15 |
```

```
+------------+
1 row in set
```

Example 4: Date is TO_ DATE() function, and TO_ The argument to the DATE() function is in date time format.

```
gbase> SELECT ADD_ MONTHS(TO_DATE('2020-3-15
12:20:31','YYYY-MM-DD HH24:MI:SS'),3) AS f_ Show FROM dual;
+---------------------+
| f_ Show             |
+---------------------+
| 2020-06-15 12:20:31 |
+---------------------+
1 row in set
```

Example 5: August 31st is the last day of August. After adding 3 months, it is November, which has a total of 30 days. Therefore, the result is "2019 November 30th".

```
gbase> SELECT ADD_
MONTHS(TO_DATE('2019-8-31','YYYY-MM-DD'),3) AS f_ Show FROM
dual;
+------------+
| f_ Show    |
+------------+
| 2019-11-30 |
+------------+
1 row in set
```

Example 6: November 30th is the last day of November. After a decrease of one month, it is October. October has a total of 31 days, so the result is "2019 October 30th".

```
gbase> SELECT ADD_
MONTHS(TO_DATE('2019-11-30','YYYY-MM-DD'),-1) AS f_ Show FROM
dual;
+------------+
| f_ Show    |
+------------+
| 2019-10-30 |
+------------+
1 row in set
```

Example 7: February 28th is the last day of February. After reducing by one month, it is January. If the default mode is 0, it returns "2020 01 28".

```
gbase> SELECT ADD_
MONTHS(TO_DATE('2020-02-28','YYYY-MM-DD'),-1) FROM dual;
+-------------------------------------------------+
```

```
| ADD_ MONTHS(TO_DATE('2020-02-28','YYYY-MM-DD'),-1) |
+----------------------------------------------------+
| 2020-01-28                                         |
+----------------------------------------------------+
1 row in set (Elapsed: 00:00:00.01)
```

If mode is 1, it returns "2020 01 31".

```
gbase> SELECT ADD_
MONTHS(TO_DATE('2020-02-28','YYYY-MM-DD'),-1,1) FROM dual;
+----------------------------------------------------+
| ADD_ MONTHS(TO_DATE('2020-02-28','YYYY-MM-DD'),-1,1) |
+----------------------------------------------------+
| 2020-01-28                                         |
+----------------------------------------------------+
1 row in set (Elapsed: 00:00:00.01)
```

5.1.5.5.5 CONVERT_ TZ(dt,from_tz,to_tz)

Function Description

CONVERT_ TZ() retrieves the datetime value dt from the_ The time zone given by tz is converted to 'to'_ Tz and return the result value. If the parameter is illegal, the function returns NULL.

If you are running from_ When converting tz to UTC, if the value exceeds the range supported by the TIMESTAMP type, the conversion will not be performed. The range of values for TIMESTAMP is described in the TIMESTAMP section.

To use names such as "MET" or "Europe/Moscow" to name time zones, a time zone table must be set appropriately.

Example

Example 1: from_ Tz is'+00:00 ', to_ Tz is' -07:00 'for time zone conversion.

```
gbase> SELECT CONVERT_ TZ('2020-01-01 12:00:00','+00:00','-07:00')
FROM dual;
+----------------------------------------------------+
| CONVERT_ TZ('2020-01-01 12:00:00','+00:00','-07:00') |
+----------------------------------------------------+
| 2020-01-01 05:00:00                                |
+----------------------------------------------------+
1 row in set
```

Example 2: CONVERT_ The TZ() function supports time zone internal conversion conversion of daylight saving time.

```
gbase> SELECT CONVERT_ TZ('2020-03-11
2:00:00','US/Eastern','US/Central') FROM dual;
+-----------------------------------------------------------+
| CONVERT_ TZ('2020-03-11 2:00:00','US/Eastern','US/Central') |
+-----------------------------------------------------------+
| 2020-03-11 01:00:00                                       |
+-----------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.02)

gbase> SELECT CONVERT_ TZ('2020-03-11
3:00:00','US/Eastern','US/Central') FROM dual;
+-----------------------------------------------------------+
| CONVERT_ TZ('2020-03-11 3:00:00','US/Eastern','US/Central') |
+-----------------------------------------------------------+
| 2020-03-11 02:00:00                                       |
+-----------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.02)

Note: The following methods do not support daylight saving time:
gbase> SELECT CONVERT_ TZ('2020-03-11 2:00:00','-6:00','-7:00') FROM
dual;
+-------------------------------------------------+
| CONVERT_ TZ('2020-03-11 2:00:00','-6:00','-7:00') |
+-------------------------------------------------+
| 2020-03-11 01:00:00                             |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.02)

gbase> SELECT CONVERT_ TZ('2020-03-11 3:00:00','-6:00','-7:00') FROM
dual;
+-------------------------------------------------+
| CONVERT_ TZ('2020-03-11 3:00:00','-6:00','-7:00') |
+-------------------------------------------------+
| 2020-03-11 02:00:00                             |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.5.6 CURDATE()

Function Description

Returns the current date value in the format "YYYY-MM-DD" or "YYYYMMDD", depending on whether the function is used in a string or numeric context.

Example

Example 1: "YYYY-MM-DD" format returns the current date.

```
gbase> SELECT CURDATE() FROM dual;
+------------+
| CURDATE()  |
+------------+
| 2020-04-01 |
+------------+
1 row in set
```

Example 2: "YYYYMMDD" format returns the current date.

```
gbase> SELECT date_ format(CURDATE(),'%Y%m%d') FROM dual;
+---------------------------------+
| date_ format(CURDATE(),'%Y%m%d') |
+---------------------------------+
| 20200401                        |
+---------------------------------+
1 row in set
```

5.1.5.5.7 CURRENT_ DATE, CURRENT_ DATE()

Function Description

CURRENT_ DATE and Current_ DATE() is equivalent to CURDATE().

Example

Example 1: Using Current_ The DATE() function returns the date.

```
gbase> SELECT CURRENT_ DATE() FROM dual;
+----------------+
| CURRENT_ DATE() |
+----------------+
| 2020-04-01     |
+----------------+
1 row in set
```

Example 2: Using Current_ The DATE variable returns the date.

```
gbase> SELECT CURRENT_ DATE FROM dual;
+--------------+
```

```
| CURRENT_ DATE    |
+---------------+
| 2020-04-01       |
+---------------+
1 row in set
```

Example 3: Using the CURDATE() function to return a date.

```
gbase> SELECT CURDATE() FROM dual;
+------------+
| CURDATE()    |
+------------+
| 2020-04-01 |
+------------+
1 row in set
```

5.1.5.5.8 CURRENT_ TIME(), CURTIME()

Function Description

CURRENT_ TIME and Current_ TIME() is equivalent to CURTIME().

Example

Example 1: Using Current_ The TIME() function returns the time.

```
gbase> SELECT CURRENT_ TIME() FROM dual;
+----------------+
| CURRENT_ TIME() |
+----------------+
| 10:47:35            |
+----------------+
1 row in set
```

Example 2: Using the CURTIME() function to return time.

```
gbase> SELECT CURTIME() FROM dual;
+-----------+
| CURTIME() |
+-----------+
| 10:48:01    |
+-----------+
1 row in set
```

5.1.5.5.9 CURRENT_ TIMESTAMP, CURRENT_ TIMESTAMP()

Function Description

CURRENT_ TIMESTAMP and CURRENT_ TIMESTAMP() is equivalent to NOW ().

Example

Example 1: Using Current_ The TIMESTAMP function returns "date+time".

```
gbase> SELECT CURRENT_ TIMESTAMP FROM dual;
+---------------------+
| CURRENT_ TIMESTAMP     |
+---------------------+
| 2020-04-01 14:23:02 |
+---------------------+
1 row in set
```

Example 2: Using Current_ The TIMESTAMP() function returns "date+time".

```
gbase> SELECT CURRENT_ TIMESTAMP() FROM dual;
+---------------------+
| CURRENT_ TIMESTAMP() |
+---------------------+
| 2020-04-01 14:23:21 |
+---------------------+
1 row in set
```

Example 3: Using the NOW () function to return "date+time".

```
gbase> SELECT NOW() FROM dual;
+---------------------+
| NOW()                        |
+---------------------+
| 2020-04-01 14:23:47 |
+---------------------+
1 row in set
```

5.1.5.5.10 DATE(expr)

Function Description

Obtain the date part from the date or datetime expression expr.

If expr is an illegal date string, return NULL.

Example

Example 1: Obtaining the date part from a datetime expression.

```
gbase> SELECT DATE('2019-09-05 11:22:03') FROM dual;
+------------------------------+
| DATE('2019-09-05 11:22:03') |
+------------------------------+
| 2019-09-05                   |
+------------------------------+
1 row in set
```

Example 2: If expr is an illegal date string, it returns NULL.

```
gbase> SELECT DATE('2019-09-32 11:22:03') FROM dual;
+------------------------------+
| DATE('2019-09-32 11:22:03') |
+------------------------------+
| NULL                         |
+------------------------------+
1 row in set, 1 warning

gbase> SHOW WARNINGS;
+-------+------+-----------------------------------------------------------------------+
| Level | Code | Message
|
+-------+------+-----------------------------------------------------------------------+
| Note    | 1292 |172.168.83.11:5050 - Incorrect datetime value: '2019-09-32
11:22:03' |
+-------+------+-----------------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.5.11 DATEDIFF(expr,expr2)

Function Description

DATEDIFF() returns the number of days between the start date expr and the end date expr2.

Expr and expr2 are date or datetime expressions. Only the date part is used for calculation. If the parameter used to calculate the date interval is not of type date or datetime, such as TIME data, the calculation result is not trustworthy.

Example

Example 1: If expr is later than expr2, the number of days returned is a positive number.

```
gbase> SELECT DATEDIFF('2019-08-30 23:59:59','2019-08-29') FROM
dual;
+---------------------------------------------+
| DATEDIFF('2019-08-30 23:59:59','2019-08-29') |
+---------------------------------------------+
|                                            1 |
+---------------------------------------------+
1 row in set
```

Example 2: expr is earlier than expr2 and returns a negative number of days.

```
gbase> SELECT DATEDIFF('2020-07-31 23:59:59','2020-08-30') FROM
dual;
+---------------------------------------------+
| DATEDIFF('2020-07-31 23:59:59','2020-08-30') |
+---------------------------------------------+
|                                          -30 |
+---------------------------------------------+
1 row in set
```

5.1.5.5.12 DATE_ ADD(), DATE_ SUB()

Function Description

The following function performs date calculation:

```
Addition operation: DATE_ ADD(date,INTERVAL expr type)
Subtraction operation: DATE_ SUB(date,INTERVAL expr type)
```

Table 520 Parameter Description

Parameter Name	Description
date	It is a DATETIME or DATE value that specifies the start of a date. Expr is an expression that specifies the time interval value to add or subtract from the start date.
dateexpr	It is a string that can start with a '-' for negative time intervals.
type	Keywords, which indicate the way the expression is interpreted.
INTERVAL	Keyword and type modifiers are case insensitive.

Table 521 Related type and expr parameters

Type value	Expected expr format
MICROSECOND	MICROSECONDS
MILLISECOND	MILLISECONDS
SECOND	SECONDS
MINUTE	MINUTES

Type value	Expected expr format
HOUR	HOURS
DAY	DAYS
WEEK	WEEKS
MONTH	MONTHS
QUARTER	QUARTERS
YEAR	YEARS
SECOND_ MICROSECOND	'SECONDS.MICROSECONDS'
MINUTE_ MICROSECOND	'MINUTES:SECONDS.MICROSECONDS'
MINUTE_ SECOND	'MINUTES:SECONDS'
HOUR_ MICROSECOND	'HOURS:MINUTES:SECONDS.MICROSECONDS'
HOUR_ SECOND	'HOURS:MINUTES:SECONDS'
HOUR_ MINUTE	'HOURS:MINUTES'
DAY_ MICROSECOND	'DAYS:HOURS:MINUTES:SECONDS.MICROSECONDS'
DAY_ SECOND	'DAYS HOURS:MINUTES:SECONDS'
DAY_ MINUTE	'DAYS HOURS:MINUTES'
DAY_ HOUR	'DAYS HOURS'
YEAR_ MONTH	'YEARS-MONTHS'

explain

- In the format of expr, GBase 8a MPP Cluster allows any character as a delimiter. The suggested delimiter characters are shown in the table. If the date parameter is a DATE value and the calculated interval only has the YEAR, MONTH, and DAY parts (no time part), then the return value is also a DATE value. Otherwise, the return value is a DATETIME value.
- If the other side of the expression is of type DATE or DATETIME, then INTERVAL expr type is allowed to appear on either side of '+'. For '-', the INTERVAL expr type value can only appear on the right because subtracting a DATE or DATETIME value from a time interval is meaningless.
- DATE_ ADD(date,INTERVAL expr type)、DATE_ When using SUB (date, INTERVAL expr type), it should be noted that when the parameter type is a composite time unit and there are other units between the two high and low units that make up the type (such as HOUR-MICROSECOND, and there are also MINITE and SECOND between HOUR and MICROSECOND), the expr parameter needs to declare the values of the high and low units and all units included in the middle in order. The expr parameter value will be filled in from the low units to the high units in order, Other units included between low and high units will not be ignored. Please strictly follow the format corresponding to type and expr in the table above. If the type is HOUR_ The expr value of MICROSECOND should be 02:00:00.000002, corresponding to HOURS: MINITES: SECONDS.MicroSECONDS.
  Example:
  select date_ add('2022-02-16 18:00:00',interval '02:00:00.000002' HOUR_MICROSECOND); The
  date has been increased by 2HOUR2MICROSECOND, and the result is: 2022-02-16 20:00:00.
  000002select date_ add('2022-02-16 18:00:00',interval '2:2' HOUR_MICROSECOND); The
  date has been added with 2SECOND2MICROSECOND, and the result is: 2022-02-16 18:00:02.00.00002
- If a user adds or subtracts a value containing time from a date type, the result will be automatically adjusted and converted to a date type.
- If the user uses an incorrect date, the return result will be NULL.
- If users add MONTH and YEAR_ MONTH or YEAR, and if the day of the result date is greater than the maximum number of days in the new month, it will be adjusted to the maximum number of days in the new month.

Example

Example 1: Add 1 second to "2020 08 30 23:59:59".

```
gbase> SELECT '2020-08-30 23:59:59' + INTERVAL 1 SECOND FROM
dual;
+-------------------------------------------+
| '2020-08-30 23:59:59' + INTERVAL 1 SECOND |
+-------------------------------------------+
| 2020-08-31 00:00:00                       |
```

```
+---------------------------------------------+
1 row in set
```

Example 2: Using DATE_ The ADD() function adds 1 second to "2020 08 30 23:59:59", and the execution result is the same as Example 1.

```
gbase> SELECT DATE_ ADD('2010-08-30 23:59:59',INTERVAL 1
SECOND) FROM dual;
+-------------------------------------------------+
| DATE_ ADD('2010-08-30 23:59:59',INTERVAL 1 SECOND) |
+-------------------------------------------------+
| 2010-08-31 00:00:00                             |
+-------------------------------------------------+
1 row in set
```

Example 3: Add 1 day to "2020 December 31 23:59:59".

```
gbase> SELECT INTERVAL 1 DAY + '2020-12-31 23:59:59' FROM dual;
+---------------------------------------+
| INTERVAL 1 DAY + '2020-12-31 23:59:59' |
+---------------------------------------+
| 2021-01-01 23:59:59                   |
+---------------------------------------+
1 row in set
```

Example 4: Using DATE_ The ADD() function adds 1 day to "2020-12-31 23:59:59", and the execution result is the same as Example 3.

```
gbase> SELECT DATE_ ADD('2020-12-31 23:59:59',INTERVAL 1 DAY)
FROM dual;
+-------------------------------------------------+
| DATE_ ADD('2020-12-31 23:59:59',INTERVAL 1 DAY) |
+-------------------------------------------------+
| 2021-01-01 23:59:59                             |
+-------------------------------------------------+
1 row in set
```

Example 5: DATE_ The ADD() function has a type of 'MINITE_SECOND'.

```
gbase> SELECT DATE_ ADD('2020-12-31 23:59:59', INTERVAL '1:1'
MINUTE_SECOND) FROM dual;
+-------------------------------------------------------------+
| DATE_ ADD('2020-12-31 23:59:59', INTERVAL '1:1' MINUTE_SECOND) |
+-------------------------------------------------------------+
| 2021-01-01 00:01:00                                         |
+-------------------------------------------------------------+
1 row in set
```

Example 6: DATE_ SUB() function with type 'DAY_SECOND'.

```
gbase> SELECT DATE_ SUB('2020-01-01 00:00:00', INTERVAL '1 1:1:1'
DAY_SECOND) FROM dual;
+-----------------------------------------------------------------+
| DATE_ SUB('2020-01-01 00:00:00', INTERVAL '1 1:1:1' DAY_SECOND) |
+-----------------------------------------------------------------+
| 2019-12-30 22:58:59                                             |
+-----------------------------------------------------------------+
1 row in set
```

Example 7: DATE_ The ADD() function has a type of 'DAY_HOUR'.

```
gbase> SELECT DATE_ ADD('2020-01-01 00:00:00', INTERVAL '-1 10'
DAY_HOUR) FROM dual;
+-----------------------------------------------------------+
| DATE_ ADD('2020-01-01 00:00:00', INTERVAL '-1 10' DAY_HOUR) |
+-----------------------------------------------------------+
| 2019-12-30 14:00:00                                       |
+-----------------------------------------------------------+
1 row in set
```

Example 8: DATE_ The ADD() function has a type of 'DAY_HOUR'.

Because '-1' subtracts 1 day, the hour '-10' is also subtracted, depending on the preceding operating symbol. The writing method in the following example is equivalent to the calculation in Example 7.

```
gbase> SELECT DATE_ ADD('2020-01-01 00:00:00', INTERVAL '-1 -10'
DAY_HOUR) FROM dual;
+-----------------------------------------------------------+
| DATE_ ADD('2020-01-01 00:00:00', INTERVAL '-1 -10' DAY_HOUR) |
+-----------------------------------------------------------+
| 2019-12-30 14:00:00                                       |
+-----------------------------------------------------------+
1 row in set
```

Example 9: DATE_ SUB() function with type 'DAY'.

```
gbase> SELECT DATE_ SUB('2020-01-02', INTERVAL 31 DAY) FROM
dual;
+---------------------------------------+
| DATE_ SUB('2020-01-02', INTERVAL 31 DAY) |
+---------------------------------------+
| 2019-12-02 00:00:00                   |
+---------------------------------------+
1 row in set
```

Example 10: DATE_ The ADD() function has a type of 'SECOND_MICROSECOND'.

```
gbase> SELECT DATE_ ADD('2020-08-31 23:59:59.000002',INTERVAL
'1.999999' SECOND_MICROSECOND) AS DATE_ ADD FROM dual;
+----------------------------+
| DATE_ ADD                  |
+----------------------------+
| 2020-09-01 00:00:01.000001 |
+----------------------------+
1 row in set
```

Example 11: Add 1 day to "2020 08 30".

```
gbase> SELECT DATE_ ADD('2020-08-30', INTERVAL 1 DAY) FROM
dual;
+----------------------------------------+
| DATE_ ADD('2020-08-30', INTERVAL 1 DAY) |
+----------------------------------------+
| 2020-08-31 00:00:00                    |
+----------------------------------------+
1 row in set
```

Example 12: Add 1 hour to the date type and convert the returned result to the "date+time" type.

```
gbase> SELECT DATE_ ADD('2020-01-01', INTERVAL 1 HOUR) FROM
dual;
+----------------------------------------+
| DATE_ ADD('2020-01-01', INTERVAL 1 HOUR) |
+----------------------------------------+
| 2020-01-01 01:00:00                    |
+----------------------------------------+
1 row in set
```

Example 13: Add January to "2020 01 30".

```
gbase> SELECT DATE_ ADD('2020-01-30', INTERVAL 1 MONTH) FROM
dual;
+----------------------------------------+
| DATE_ ADD('2020-01-30', INTERVAL 1 MONTH) |
+----------------------------------------+
| 2020-02-29 00:00:00                    |
+----------------------------------------+
1 row in set
```

5.1.5.5.13 DATE_ FORMAT(date,FORMAT)

Function Description

Format the date value according to the FORMAT string.

The following format can be used in format strings:

Table 522 Format Description

Format	Description
%a	English abbreviation for Sunday name (Sun... Sat)
%b	English abbreviation for month (Jan... DEC)
%c	The numerical form of the month (0... 12)
%D	The day of a month with an English suffix (0th, 1st, 2nd, 3rd...)
%d	Number of days in the month, in numerical form (00... 31)
%e	Number of days in the month, in numerical form (0... 31)
%f	Microseconds (000000... 999999)
%H	Hour, 24-hour system (00... 23)
%h	Hour, 12 hour system (0,1... 12)
%I	Hour, 12 hour system, with a single digit preceded by 0 (01... 12)
%i	Minute, in numerical form (00... 59)
%j	Number of days in a year (001... 366)
%k	Hour, 24-hour system (0... 23)
%l	Hour, 12 hour system (1... 12)
%M	January... December
%m	Month, in numerical form (00... 12)
%p	AM or PM
%r	Time, 12 hour format (HH: MI: SS followed by AM or PM)
%S	Seconds (00... 59)
%s	Seconds (00... 59)
%T	Time, 24 hours (HH: MI: SS)
%U	Week (00... 53), Sunday is the first day of the week
%u	Monday is the first day of the week (00... 53)
%V	Week (01... 53), Sunday is the first day of the week Used with '% X'
%v	Monday (01... 53) is the first day of a week Used with '% x'
%W	Sunday... Saturday
%w	Which day of the week (0=Sunday... 6=Saturday)
%X	Reflect the year of the week in 4-digit form
%x	Reflect the year of the week in 4-digit form
%Y	Year expressed in 4-digit numerical form
%y	Year expressed in 2-digit numerical form

Format	Description
%%	One character '%'
%.	One or more characters other than letters, numbers, and spaces
%@	One or more letters
%#	One or more numbers

All other characters are copied directly into the result without explanation.

be careful

The '%' character is required before the format specification.

Example

Example 1: The Format format is'% W% M% Y '.

```
gbase> SELECT DATE_ FORMAT('2020-10-04 22:23:00', '%W %M %Y')
FROM dual;
+-----------------------------------------------+
| DATE_ FORMAT('2020-10-04 22:23:00', '%W %M %Y') |
+-----------------------------------------------+
| Sunday October 2020                           |
+-----------------------------------------------+
1 row in set
```

Example 2: The Format format is'% H:% i:% s'.

```
gbase> SELECT DATE_ FORMAT('2020-10-04 22:23:00', '%H:%i:%s')
FROM dual;
+-----------------------------------------------+
| DATE_ FORMAT('2020-10-04 22:23:00', '%H:%i:%s') |
+-----------------------------------------------+
| 22:23:00                                      |
+-----------------------------------------------+
1 row in set
```

Example 3: The Format format is'% D% y% a% d% m% b% j '.

```
gbase> SELECT DATE_ FORMAT('2020-10-04
22:23:00','%D %y %a %d %m %b %j') FROM dual;
+-------------------------------------------------------+
| DATE_ FORMAT('2020-10-04 22:23:00','%D %y %a %d %m %b %j') |
+-------------------------------------------------------+
| 4th 20 Sun 04 10 Oct 278                              |
+-------------------------------------------------------+
1 row in set
```

Example 4: The Format format is'% H% k% I% r% T% S% w '.

```
gbase> SELECT DATE_ FORMAT('2020-10-04
22:23:00','%H %k %I %r %T %S %w') FROM dual;
+-----------------------------------------------------------+
| DATE_ FORMAT('2020-10-04 22:23:00','%H %k %I %r %T %S %w') |
+-----------------------------------------------------------+
| 22 22 10 10:23:00 PM 22:23:00 00 0                        |
+-----------------------------------------------------------+
1 row in set
```

Example 5: The Format format is'% X% V '.

```
gbase> SELECT DATE_ FORMAT('2020-01-01', '%X %V') FROM dual;
+-------------------------------------+
| DATE_ FORMAT('2020-01-01', '%X %V') |
+-------------------------------------+
| 2019 52                             |
+-------------------------------------+
1 row in set
```

5.1.5.5.14 DATE_ BIN

Function Description

This function "boxes" the input timestamp into the specified interval and aligns it with the specified timestamp.

Allow expr to be a negative interval, treat it as a positive interval, and the interval cannot contain months or larger units.

Alignment rules:

- When the origin plus/minus n interval expr types can be closest to and less than or equal to source, it is considered aligned.
- When origin<source, add interval expr type.
- When origin>source, subtract interval expr type.

be careful

Cross time zone calculation is not supported.

Grammar format

```
DATE_ BIN(INTERVAL expr type, source, origin)
```

Table 597 Parameter Description

Parameter Name	explain
INTERVAL expr type	The time interval value used to specify the interval, aligning source with origin based on the interval specified by expr. Expr is a string that represents the time interval for negative values. It can start with a '-'. Type is the keyword that indicates how expr is interpreted. The INTERVAL keyword and type modifier are case insensitive.
source	It is a datetime type and will automatically be converted to a datetime type when the type is date.
origin	It is a datetime type and will automatically be converted to a datetime type when the type is date.

The relevant type and expr parameters in Table -597 are as follows

Type value	Expected expr format
SECOND	SECONDS
MINUTE	MINUTES
HOUR	HOURS
WEEK	WEEKS
DAY	DAYS
DAY_ HOUR	'DAYS HOURS'
DAY_ MINUTE	'DAYS HOURS:MINUTES'
DAY_ SECOND	'DAYS HOURS:MINUTES:SECONDS'
HOUR_ MINUTE	'HOURS:MINUTES'
HOUR_ SECOND	'HOURS:MINUTES:SECONDS'
MINUTE_ SECOND	'MINUTES:SECONDS'
SECOND_ MICROSECOND	'SECONDS.MICROSECONDS'
MINUTE_ MICROSECOND	'MINUTES.MICROSECONDS'
HOUR_ MICROSECOND	'HOURS.MICROSECONDS'
DAY_ MICROSECOND	'DAYS.MICROSECONDS'

Example

```
SELECT DATE_ BIN(interval 15 MINUTE, '2020-02-11 15:44:17','2001-01-01');

+-----------------------------------------------------------------------------------+
| DATE_ BIN(interval 15 MINUTE, '2020-02-11 15:44:17','2001-01-01') |
+-----------------------------------------------------------------------------------+
| 2020-02-11 15:30:00                                                               |
+-----------------------------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

DATE_ BIN(interval 3 MICROSECOND, '2020-02-11 15:44:17.1','2001-01-01 12:44:17');
2020-02-11 15:44:17.099999
DATE_ BIN(interval 3 MILLISECOND, '2020-02-11 15:44:17.1','2001-01-01 12:44:17');
2020-02-11 15:44:17.099

SELECT DATE_ BIN(interval '1:1' MINUTE_SECOND, '2020-02-11 15:44:17','2001-01-01
12:44:17');

+-----------------------------------------------------------------------------------------------------------+
| DATE_ BIN(interval '1:1' MINUTE_SECOND, '2020-02-11 15:44:17','2001-01-01 12:44:17') |
+-----------------------------------------------------------------------------------------------------------+
|2020-02-11 15:44:00                                                                                        |
+-----------------------------------------------------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

SELECT DATE_ BIN(interval 12 DAY,'2020-02-11 15:44:17', '2001-01-01 12:44:17');
+-----------------------------------------------------------------------------------+
| DATE_ BIN(interval 12 DAY,'2020-02-11 15:44:17', '2001-01-01 12:44:17') |
+-----------------------------------------------------------------------------------+
| 2020-02-03 12:44:17                                                               |
+-----------------------------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.5.15 DATE_ PART

Function Description

date_ The part (type, date) function implements the interception of date according to the specified type and outputs the obtained results. The function is the same as extract, but the syntax is different and the supported types are different.

explain

date_ The part function needs to be equivalent to the extract function to achieve syntax format compatibility, extract sub domains from date/time values, and add new sub domains.

Grammar format

```
date_ part(type,date)
```

Table 597 Parameter Description

Parameter Name	explain
type	Used to specify the time portion to be truncated, and retrieve the time from the date based on the type. The types supported by type are keywords, which indicate how date is intercepted.
date	An expression of type 'datetime timestamp' will automatically be converted to a 'datetime' type when the type is of another type to evaluate the function value. Other input values or types will also be processed in the same way as being converted to a 'datetime' type. If the conversion is incorrect, just like other implicit conversions, a warning will be reported.

For better compatibility with postgreSQL and netezza database function usage, date_ The type of the part function supports both the keyword method of GBase8a and the string method of the two databases to input the corresponding type.

The following types also apply to the extract function.

GBase8a Writing Method	Compatible writing	explain
Year	'Year'	year
Year_ month	'Year_ month'	years
Quarter	'Quarter'	quarter
Month	'Month'	month
Week	'Week'	week
Day	'Day'	date
Day_ hour	'Day_ hour'	Days and hours
Day_ minute	'Day_ minute'	Days, hours, minutes

GBase8a Writing Method	Compatible writing	explain
Day_ second	'Day_ second'	Days, hours, minutes, seconds
Hour	'Hour'	hour
Hour_ minute	'Hour_ minute'	Hours and minutes
Hour_ second	'Hour_ second'	Hours, minutes, seconds
Minute	'Minute'	minute
Minute_ second	'Minute_ second'	Minutes seconds
Second	'Second'	second
Millisecond	'Millisecond'	millisecond
Microsecond	'Microsecond'	Microsecond
Day_ microsecond	'Day_ microsecond'	Days, hours, minutes, seconds, milliseconds
Hour_ microsecond	'Hour_ microsecond'	Hours, minutes, seconds, milliseconds
Minute_ microsecond	'Minute_ microsecond'	Minutes, seconds, milliseconds
Second_ microsecond	'Second_ microsecond'	Seconds and milliseconds
Century	'Century'	century
Decade	'Decade'	decade
Dow	'Dow'	Day of the week
Doy	'Doy'	Day of the year
Epoch	'Epoch'	Same as Unix_ Timestamp function, with an additional microsecond bit
Isodow	'Isodow'	Days of the week, from Monday (1) to Sunday (7)
Isoyear	'Isoyear'	Date falls in the year of ISO 8601 week number
Millennium	'Millennium'	Millennium

GBase8a Writing Method	Compatible writing	explain
Millisecnds	'Millisecnds'	Seconds domain, milliseconds
Microseconds	'Microseconds'	Second domain_ microsecond

be careful

Date_ The part (type, date) function returns a result of decimal when the type is Milliseconds. The return type for other scenarios is bigint (12).

Epoch on date_ The part function is not supported, only the extract functi on is supported.

Example

Example: Date truncation function date_ part。

```
gbase> select date_ part('Year','2014-06-28 11:12:13.45678');
+-------------------------------------------+
| date_ part('Year','2014-06-28 11:12:13.45678') |
+-------------------------------------------+
|                                      2014 |
+-------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('Year_month','2014-06-28 11:12:13.45678');
+-----------------------------------------------+
| date_ part('Year_month','2014-06-28 11:12:13.45678') |
+-----------------------------------------------+
|                                        201406 |
+-----------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('quarter','2014-06-28 11:12:13.45678');
+-------------------------------------------------+
| date_ part('quarter','2014-06-28 11:12:13.45678') |
+-------------------------------------------------+
|                                               2 |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('month','2014-06-28 11:12:13.45678');
```

```
+-----------------------------------------------+
| date_ part('month','2014-06-28 11:12:13.45678') |
+-----------------------------------------------+
|                                             6 |
+-----------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('week','2014-06-28 11:12:13.45678');
+-----------------------------------------------+
| date_ part('week','2014-06-28 11:12:13.45678') |
+-----------------------------------------------+
|                                            25 |
+-----------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('day','2014-06-28 11:12:13.45678');
+-----------------------------------------------+
| date_ part('day','2014-06-28 11:12:13.45678') |
+-----------------------------------------------+
|                                            28 |
+-----------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('Millisecond','2014-06-28 11:12:13.45678');
+-------------------------------------------------+
| date_ part('Millisecond','2014-06-28 11:12:13.45678') |
+-------------------------------------------------+
|                                             456 |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('Microsecond','2014-06-28 11:12:13.45678');
+-------------------------------------------------+
| date_ part('Microsecond','2014-06-28 11:12:13.45678') |
+-------------------------------------------------+
|                                          456780 |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('Day_Microsecond','2014-06-28 11:12:13.45678');
+-----------------------------------------------------+
| date_ part('Day_Microsecond','2014-06-28 11:12:13.45678') |
+-----------------------------------------------------+
|                                      28111213456780 |
```

```
+-----------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('century','2014-06-28 11:12:13.45678');
+-------------------------------------------------+
| date_ part('century','2014-06-28 11:12:13.45678') |
+-------------------------------------------------+
|                                              21 |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('decade','2014-06-28 11:12:13.45678');
+------------------------------------------------+
| date_ part('decade','2014-06-28 11:12:13.45678') |
+------------------------------------------------+
|                                             201 |
+------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('Dow','2014-06-28 11:12:13.45678');
+-----------------------------------------------+
| date_ part('Dow','2014-06-28 11:12:13.45678') |
+-----------------------------------------------+
|                                             6 |
+-----------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('Doy','2014-06-28 11:12:13.45678');
+-----------------------------------------------+
| date_ part('Doy','2014-06-28 11:12:13.45678') |
+-----------------------------------------------+
|                                           179 |
+-----------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('Isodow','2014-06-28 11:12:13.45678');
+-------------------------------------------------+
| date_ part('Isodow','2014-06-28 11:12:13.45678') |
+-------------------------------------------------+
|                                              6 |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('Isoyear','2014-06-28 11:12:13.45678');
```

```
+-------------------------------------------------+
| date_ part('Isoyear','2014-06-28 11:12:13.45678') |
+-------------------------------------------------+
|                                            2014 |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('Millennium','2014-06-28 11:12:13.45678');
+-------------------------------------------------+
| date_ part('Millennium','2014-06-28 11:12:13.45678') |
+-------------------------------------------------+
|                                               3 |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('Milliseconds','2014-06-28 11:12:13.45678');
+-------------------------------------------------+
| date_ part('Milliseconds','2014-06-28 11:12:13.45678') |
+-------------------------------------------------+
|                                        13456.78 |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select date_ part('Microseconds','2014-06-28 11:12:13.45678');
+-------------------------------------------------+
| date_ part('Microseconds','2014-06-28 11:12:13.45678') |
+-------------------------------------------------+
|                                        13456780 |
+-------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.5.16 DAY(date)

Function Description

The return date is the day ordinal of a month, ranging from 1 to 31.

Example

Example 1: Returns the day in mid August when "2020 08 30" is the day.

```
gbase> SELECT DAY('2020-08-30'),DAYOFMONTH('2020-08-30') FROM
dual;
```

```
+-------------------+--------------------------+
| DAY('2020-08-30') | DAYOFMONTH('2020-08-30') |
+-------------------+--------------------------+
|                30 |                       30 |
+-------------------+--------------------------+
1 row in set
```

5.1.5.5.17 DAYNAME(date)

Function Description

Returns the given date, which is the day of the week.

Example

Example 1: Returns the day of the week for "2020 08 30".

```
gbase> SELECT DAYNAME('2020-08-30') FROM dual;
+-----------------------+
| DAYNAME('2020-08-30') |
+-----------------------+
| Sunday                |
+-----------------------+
1 row in set
```

5.1.5.5.18 DAYOFMONTH(date)

Function Description

The return date is the day ordinal of a month, ranging from 1 to 31.

DayOFMONTH() is equivalent to Day().

Example

Example 1: Returns the day in mid August when "2020 08 30" is the day.

```
gbase> SELECT DAYOFMONTH('2020-08-30') FROM dual;
+--------------------------+
| DAYOFMONTH('2020-08-30') |
+--------------------------+
|                       30 |
+--------------------------+
1 row in set
```

Example 2: Returns the day of November 2020-11-00.

```
gbase> SELECT DAYOFMONTH('2020-11-00') FROM dual;
+--------------------------+
| DAYOFMONTH('2020-11-00') |
+--------------------------+
|                     NULL |
+--------------------------+
1 row in set, 1 warning (Elapsed: 00:00:00.02)

gbase> show warnings;
+-------+------+-----------------------------------------------------------------+
| Level | Code | Message
|
+-------+------+-----------------------------------------------------------------+
| Note  | 1292 | 172.168.63.11:5050 - Incorrect datetime value: '2020-11-00' |
+-------+------+-----------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.5.19 DAYOFWEEK(date)

Function Description

Returns the weekday index corresponding to date (1=Sunday, 2=Monday,..., 7=Saturday).

Example

Example 1: "2020 08 30" is a Sunday, and the corresponding weekday index is returned as 1.

```
gbase> SELECT DAYOFWEEK('2020-08-30') FROM dual;
+--------------------------+
| DAYOFWEEK('2020-08-30') |
+--------------------------+
|                        1 |
+--------------------------+
1 row in set
```

5.1.5.5.20 DAYOFYEAR(date)

Function Description

The return date is the day of the year, ranging from 1 to 366.

Example

Example 1: Returns the day of 2020 when "2020 08 30" is the year.

```
gbase> SELECT DAYOFYEAR('2020-08-30') FROM dual;
+--------------------------+
| DAYOFYEAR('2020-08-30') |
+--------------------------+
|                      243 |
+--------------------------+
1 row in set
```

Example 2: Returns the day of 2020 when "2020 December 31" is the year.

```
gbase> SELECT DAYOFYEAR('2020-12-31') FROM dual;
+--------------------------+
| DAYOFYEAR('2020-12-31') |
+--------------------------+
|                      366 |
+--------------------------+
1 row in set
```

5.1.5.5.21 EXTRACT(type FROM date)

Function Description

Using the TRACT() function with DATE_ ADD() or DATE_ SUB() has a consistent interval type, but it is used to specify the part extracted from the date, rather than performing date arithmetic operations.

The following table shows the types that can be returned, and type types can be used in combination.

Table -523 Types of Returnable Types

appointment	explain
year	year
quarter	quarter
month	month
day	day
week	week
hour	hour
minute	minute
second	second
Microsecond	microsecond
Century (1-100)	CENTURY
Ten years (year/10)	DECADE
Days of the week, from Sunday (0) to	DOW

appointment	explain
Saturday (6)	
Day of the year (1-365/366)	DOY
Seconds since 1970-01-01 00:00:00 UTC, with loss, only supported within the timestamp range	EPOCH
Days of the week, from Monday (1) to Sunday (7)	ISODOW
Date falls in the year of ISO 8601 week number	ISOYEAR
Microseconds. Second field, including decimal parts * 1000000	MICROSECONDS
A thousand years. Which millennium is the year in (millennium+1)	MILLENNIUM
millisecond. Second field, including decimal parts * 1000	MILLSECONDS

Example

Example 1: The returned result is the "year" in the date.

```
gbase> SELECT EXTRACT(YEAR FROM '2020-08-30') FROM dual;
+----------------------------------+
| EXTRACT(YEAR FROM '2020-08-30') |
+----------------------------------+
|                             2020 |
+----------------------------------+
1 row in set
```

Example 2: The returned result is the "year and year" in the date.

```
gbase> SELECT EXTRACT(YEAR_MONTH FROM '2020-08-30 01:02:03')
FROM dual;
+-------------------------------------------------+
| EXTRACT(YEAR_MONTH FROM '2020-08-30 01:02:03') |
+-------------------------------------------------+
|                                          202008 |
+-------------------------------------------------+
1 row in set
```

Example 3: The returned result is the part of the day, hour, and minute in the date.

```
gbase> SELECT EXTRACT(DAY_MINUTE FROM '2020-08-30 01:02:03')
FROM dual;
+-----------------------------------------------+
| EXTRACT(DAY_MINUTE FROM '2020-08-30 01:02:03') |
+-----------------------------------------------+
```

```
|                                                  300102 |
+-------------------------------------------------+
1 row in set
```

Example 4: The returned result is "1230", representing 1230 microseconds.

```
gbase> SELECT EXTRACT(MICROSECOND FROM '2020-08-30
10:30:00.00123') FROM dual;
+-----------------------------------------------------+
| EXTRACT(MICROSECOND FROM '2020-08-30 10:30:00.00123') |
+-----------------------------------------------------+
|                                                 1230 |
+-----------------------------------------------------+
1 row in set
```

Example 5: Return result "27", representing the 27th week of 2020.

```
gbase> SELECT EXTRACT(WEEK FROM '2020-07-05 10:30:00.00123')
FROM dual;
+-----------------------------------------------+
| EXTRACT(WEEK FROM '2020-07-05 10:30:00.00123') |
+-----------------------------------------------+
|                                            27 |
+-----------------------------------------------+
1 row in set
```

5.1.5.5.22 FROM_ DAYS(N)

Function Description

Returns the DATE value corresponding to the number of days N.

Example

Example 1: The corresponding DATE for '737881' is' 2020 04 01 '.

```
gbase> SELECT FROM_ DAYS(737881) FROM dual;
+------------------+
| FROM_ DAYS(737881) |
+------------------+
| 2020-04-01       |
+------------------+
1 row in set
```

5.1.5.5.23 FROM_ UNIXTIME()

grammar

```
FROM_ UNIXTIME(unix_timestamp)
FROM_ UNIXTIME(unix_timestamp,FORMAT)
```

Function Description

Returns a unix in the format 'YYYY-MM-DD HH: MI: SS' or' YYYYMMDDHHMISS'_ The timestamp parameter value, the form of the returned value depends on whether it is used in a string or a number.

If FORMAT has been given, the format of the return value follows the format of the FORMAT string. FORMAT can include the same as DATE_ The same modifier as the FORMAT() function.

Example

Example 1: Returns a date time value in the format of "YYYY-MM-DD HH: MI: SS".

```
gbase> SELECT FROM_ UNIXTIME(1585736116) FROM dual;
+---------------------------+
| FROM_ UNIXTIME(1585736116) |
+---------------------------+
| 2020-04-01 18:15:16       |
+---------------------------+
1 row in set
```

Example 2: The format is'% Y% D% M% h:% i:% s% x. '.

```
gbase> SELECT FROM_
UNIXTIME(UNIX_TIMESTAMP(),'%Y %D %M %h:%i:%s %x') FROM
dual;
+-----------------------------------------------------+
| FROM_ UNIXTIME(UNIX_TIMESTAMP(),'%Y %D %M %h:%i:%s %x') |
+-----------------------------------------------------+
| 2020 1st April 06:16:54 2020                        |
+-----------------------------------------------------+
1 row in set
```

5.1.5.5.24 GET_ FORMAT()

grammar

GET_ FORMAT(DATE|TIME|DATETIME,EUR'|'USA'|'JIS'|'ISO'|'INTERNAL')

Returns a format string.

Function Description

This function can be combined with DATE_ FORMAT() function or STR_ TO_ The DATE() function is combined.

For parameters DATE, DATETIME, and TIME, there are five possible values each, totaling fifteen format strings:

Table -524 Function Calls and String Formats

function call	result
GET_ FORMAT(DATE,'USA')	'%m.% d.%Y'
GET_ FORMAT(DATE,'JIS')	'%Y-%m-%d'
GET_ FORMAT(DATE,'ISO')	'%Y-%m-%d'
GET_ FORMAT(DATE,'EUR')	'%d.% m.%Y'
GET_ FORMAT(DATE,'INTERNAL')	'%Y%m%d'
GET_ FORMAT(DATETIME,'USA')	'%Y-%m-%d-%H.% i.%s'
GET_ FORMAT(DATETIME,'JIS')	'%Y-%m-%d %H:%i:%s'
GET_ FORMAT(DATETIME,'ISO')	'%Y-%m-%d %H:%i:%s'
GET_ FORMAT(DATETIME,'EUR')	'%Y-%m-%d-%H.% i.%s'
GET_ FORMAT(DATETIME,'INTERNAL')	'%Y%m%d%H%i%s'
GET_ FORMAT(TIME,'USA')	'%h:%i:%s %p'
GET_ FORMAT(TIME,'JIS')	'%H:%i:%s'
GET_ FORMAT(TIME,'ISO')	'%H:%i:%s'
GET_ FORMAT(TIME,'EUR')	'%H.% i.%S'
GET_ FORMAT(TIME,'INTERNAL')	'%H%i%s'

explain

For the functions of the descriptors used in the above table, please refer to the table in "5.1.5.5.13 DATE_FORMAT (date, FORMAT)".

Example

Example 1: DATE_ FORMAT() and GET_ The FORMAT() function is combined.

GET_ The corresponding output format for Format (DATE, 'EUR') is'% d.% m.% Y. '.

```
gbase> SELECT DATE_
FORMAT('2020-08-30',GET_FORMAT(DATE,'EUR')) FROM dual;
+-----------------------------------------------+
| DATE_ FORMAT('2020-08-30',GET_FORMAT(DATE,'EUR')) |
```

```
+--------------------------------------------------+
| 30.08.2020                                       |
+--------------------------------------------------+
1 row in set
```

Example 2: STR_ TO_ DATE() and GET_ The FORMAT() function is combined.

GET_ The output format corresponding to FORM (DATE, 'USA') is'% Y -% m -% d '.

```
gbase> SELECT STR_ TO_
DATE('08.30.2020',GET_FORMAT(DATE,'USA')) FROM dual;
+--------------------------------------------------+
| STR_ TO_ DATE('08.30.2020',GET_FORMAT(DATE,'USA')) |
+--------------------------------------------------+
| 2020-08-30                                       |
+--------------------------------------------------+
1 row in set
```

5.1.5.5.25 HOUR(time)

Function Description

Returns the hour value corresponding to time, with a return value range of 0 to 23 for hour values.

Example

Example 1: Returns the hour value corresponding to "10:05:03".

```
gbase> SELECT HOUR('10:05:03') FROM dual;
+------------------+
| HOUR('10:05:03') |
+------------------+
|                10 |
+------------------+
1 row in set
```

Example 2: The time value range is actually large, so the HOUR return value can be larger than 23.

```
gbase> SELECT HOUR('272:59:59') FROM dual;
+------------------+
| HOUR('272:59:59') |
+------------------+
|               272 |
+------------------+
1 row in set
```

5.1.5.5.26 LAST_ DAY(date)

Function Description

Returns the value of the last day corresponding to the current month in date. Among them, date is a date or date time type. If the parameter date is invalid, it returns NULL.

Example

Example 1: The date value is a valid date and returns the last day of August 2020.

```
gbase> SELECT LAST_ DAY('2020-08-30') FROM dual;
+------------------------+
| LAST_ DAY('2020-08-30') |
+------------------------+
| 2020-08-31             |
+------------------------+
1 row in set
```

Example 2: The date value is a valid date and returns the last day of February 2020.

```
gbase> SELECT LAST_ DAY('2020-02-05') FROM dual;
+------------------------+
| LAST_ DAY('2020-02-05') |
+------------------------+
| 2020-02-29             |
+------------------------+
1 row in set
```

Example 3: The date value is a valid date of the date time type, returning the last day of January 2020.

```
gbase> SELECT LAST_ DAY('2020-01-01 01:01:01') FROM dual;
+---------------------------------+
| LAST_ DAY('2020-01-01 01:01:01') |
+---------------------------------+
| 2020-01-31                      |
+---------------------------------+
1 row in set
```

Example 4: The date value is an invalid date and the return result is NULL.

```
gbase> SELECT LAST_ DAY('2020-08-32') FROM dual;
+------------------------+
| LAST_ DAY('2020-08-32') |
+------------------------+
| NULL                   |
```

```
+-------------------------+
1 row in set, 1 warning (Elapsed: 00:00:00.02)
gbase> SHOW WARNINGS;
+-------+------+-----------------------------------------------------------------+
| Level | Code | Message
|
+-------+------+-----------------------------------------------------------------+
| Note    | 1292 | 172.168.83.11:5050 - Incorrect datetime value: '2020-08-32' |
+-------+------+-----------------------------------------------------------------+2 row in set
```

5.1.5.5.27 LOCALTIME，LOCALTIME()

Function Description

LOCALTIME and LOCALTIME() are equivalent to NOW.

Example

Example 1: Use the LOCALTIME function to return the current "date+time".

```
gbase> SELECT LOCALTIME FROM dual;
+---------------------+
| LOCALTIME                    |
+---------------------+
| 2020-04-01 18:22:09 |
+---------------------+
1 row in set
```

Example 2: Use the LOCALTIME() function to return the current "date+time".

```
gbase> SELECT LOCALTIME() FROM dual;
+---------------------+
| LOCALTIME()                |
+---------------------+
| 2020-04-01 18:22:23 |
+---------------------+
1 row in set
```

Example 3: Use the NOW () function to return the current "date+time".

```
gbase> SELECT NOW() FROM dual;
+---------------------+
| NOW()                        |
+---------------------+
| 2020-04-01 18:22:36 |
+---------------------+
1 row in set
```

5.1.5.5.28 LOCALTIMESTAMP，LOCALTIMESTAMP()

Function Description

LOCALTIMESTAMP and LOCALTIMESTAMP() are equivalent to NOW.

Example

Example 1: Use the LOCALTIMESTAMP function to return the current timestamp.

```
gbase> SELECT LOCALTIMESTAMP FROM dual;
+---------------------+
| LOCALTIMESTAMP      |
+---------------------+
| 2020-04-01 18:22:59 |
+---------------------+
1 row in set
```

Example 2: Use the LOCALTIMESTAMP() function to return the current timestamp.

```
gbase> SELECT LOCALTIMESTAMP() FROM dual;
+---------------------+
| LOCALTIMESTAMP()    |
+---------------------+
| 2020-04-01 18:23:13 |
+---------------------+
1 row in set
```

Example 3: Use the NOW () function to return the current "date+time".

```
gbase> SELECT NOW() FROM dual;
+---------------------+
| NOW()               |
+---------------------+
| 2020-04-01 18:23:32 |
+---------------------+
1 row in set
```

5.1.5.5.29 MAKEDATE(year,dayofyear)

Function Description

Provide the year value and the day value of the year, dayof year, and return the date value.

Dayof year must be greater than 0, otherwise it returns NULL.

Example

Example 1: Returns the dates corresponding to the 31st and 32nd days of 2020.

```
gbase> SELECT MAKEDATE(2020,31), MAKEDATE(2020,32) FROM dual;
+------------------+------------------+
| MAKEDATE(2020,31) | MAKEDATE(2020,32) |
+------------------+------------------+
| 2020-01-31        | 2020-02-01       |
+------------------+------------------+
1 row in set
```

Example 2: Returns the dates corresponding to the 365th day of 2020 and 2021.

```
gbase> SELECT MAKEDATE(2020,365), MAKEDATE(2021,365) FROM
dual;
+------------------+------------------+
| MAKEDATE(2020,365) | MAKEDATE(2021,365) |
+------------------+------------------+
| 2020-12-30        | 2021-12-31       |
+------------------+------------------+
1 row in set
```

Example 3: If the dayof year value is equal to 0, it returns NULL.

```
gbase> SELECT MAKEDATE(2020,0) FROM dual;
+----------------+
| MAKEDATE(2020,0) |
+----------------+
| NULL           |
+----------------+
1 row in set
```

5.1.5.5.30 MAKETIME(hour,minute,second)

Function Description

Returns the time value calculated from hour, minute, and second.

Example

Example 1: Returns the value of the hour, minute, and second corresponding to "12,15,30".

```
gbase> SELECT MAKETIME(12,15,30) FROM dual;
+-----------------+
```

```
| MAKETIME(12,15,30) |
+--------------------+
| 12:15:30           |
+--------------------+
1 row in set
```

5.1.5.5.31 MICROSECOND(expr)

Function Description

Returns the microsecond value in the time or datetime expression expr in the form of a number, ranging from 0 to 999999.

Example

Example 1: Returns the microsecond value in time.

```
gbase> SELECT MICROSECOND('12:00:00.123456') FROM dual;
+--------------------------------+
| MICROSECOND('12:00:00.123456') |
+--------------------------------+
|                         123456 |
+--------------------------------+
1 row in set
```

Example 2: Returns the microsecond value in the datetime.

```
gbase> SELECT MICROSECOND('2019-12-31 23:59:59.000010') FROM
dual;
+-----------------------------------------+
| MICROSECOND('2019-12-31 23:59:59.000010') |
+-----------------------------------------+
|                                        10 |
+-----------------------------------------+
1 row in set
```

Example 3: If the microsecond value in the datetime exceeds six digits, the result returns the first six digits.

```
gbase> SELECT MICROSECOND('2019-12-31 23:59:59.1234567') FROM
dual;
+-----------------------------------------+
| MICROSECOND('2019-12-31 23:59:59.1234567') |
+-----------------------------------------+
|                                    123456 |
+-----------------------------------------+
```

```
1 row in set, 1 warnings

gbase> SHOW WARNINGS;
+-------+------+-----------------------------------------------------------------------------------
-+
| Level | Code | Message
|
+-------+------+-----------------------------------------------------------------------------------
-+
| Note  | 1292 | 172.168.83.11:5050 - Truncated incorrect time value: '2019-12-31
23:59:59.1234567' |
+-------+------+-----------------------------------------------------------------------------------
-+
1 row in set
```

5.1.5.5.32 MINUTE(time)

Function Description

Returns the minute value corresponding to time, ranging from 0 to 59.

Example

Example 1: The minute value corresponding to the time value is within 0-59.

```
gbase> SELECT MINUTE('01-02-03 10:08:03') FROM dual;
+----------------------------+
| MINUTE('01-02-03 10:08:03') |
+----------------------------+
|                          8 |
+----------------------------+
1 row in set
```

Example 2: If the minute value corresponding to the time value is greater than 59, it returns NULL.

```
gbase> SELECT MINUTE('01-02-03 10:60:03') FROM dual;
+----------------------------+
| MINUTE('01-02-03 10:60:03') |
+----------------------------+
|                       NULL |
+----------------------------+
1 row in set, 2 warnings

gbase> SHOW WARNINGS;
```

```
+-------+------+----------------------------------------------------------------------+
| Level | Code | Message
|
+-------+------+----------------------------------------------------------------------+
| Note  | 1292 | 172.168.83.11:5050 - Truncated incorrect time value: '01-02-03
10:60:03' |
+-------+------+----------------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.5.33 MONTH(date)

Function Description

Returns the corresponding month value in date, ranging from 1 to 12.

Example

Example 1: The month value corresponding to the date value is within 1-12.

```
gbase> SELECT MONTH('2020-08-30') FROM dual;
+---------------------+
| MONTH('2020-08-30') |
+---------------------+
|                   8 |
+---------------------+
1 row in set
```

Example 2: If the month value corresponding to the date value is greater than 12, it returns NULL.

```
gbase> SELECT MONTH('2020-13-30') FROM dual;
+---------------------+
| MONTH('2020-13-30') |
+---------------------+
|                NULL |
+---------------------+
1 row in set, 1 warning
```

5.1.5.5.34 MONTHNAME(date)

Function Description

The corresponding month in date is returned and displayed in English.

Example

Example 1: Return the English name corresponding to August.

```
gbase> SELECT MONTHNAME('2020-08-30') FROM dual;
+--------------------------+
| MONTHNAME('2020-08-30') |
+--------------------------+
| August                   |
+--------------------------+
1 row in set
```

5.1.5.5.35 NOW()

Function Description

Returns the current date time value in the format "YYYY-MM-DD HH: MI: SS" or "YYYYMMDDHHMISS", depending on whether the function is used in a string or numeric context.

Example

Example 1: Returns a date time value in the format of "YYYY-MM-DD HH: MI: SS".

```
gbase> SELECT NOW() FROM dual;
+---------------------+
| NOW()               |
+---------------------+
| 2020-04-01 18:28:56 |
+---------------------+
1 row in set
```

5.1.5.5.36 PERIOD_ ADD(P,N)

Function Description

Add N months to cycle P, and the return value format is YYYYMM.

Among them, the format of P is YYMM or YYYYMM.

be careful

Parameter P is not a numerical value of date type.

Example

Example 1: Add 2 months to "1908".

```
gbase> SELECT PERIOD_ ADD(1908,2) FROM dual;
+--------------------+
| PERIOD_ ADD(1908,2) |
+--------------------+
|                201910 |
+--------------------+
1 row in set
```

Example 2: Add 2 months to "202008".

```
gbase> SELECT PERIOD_ ADD(202008,2) FROM dual;
+----------------------+
| PERIOD_ ADD(202008,2) |
+----------------------+
|                 202010 |
+----------------------+
1 row in set
```

5.1.5.5.37 PERIOD_ DIFF(P1,P2)

Function Description

Returns the number of months between P1 and P2.

The formats of P1 and P2 are specified in YYMM or YYYYMM.

be careful

Parameters P1 and P2 are not date values.

Example

Example 1: Returns the number of months between "2002" and "201903".

```
gbase> SELECT PERIOD_ DIFF(2002,201903) FROM dual;
+--------------------------+
| PERIOD_ DIFF(2002,201903) |
+--------------------------+
|                        11 |
+--------------------------+
1 row in set
```

5.1.5.5.38 QUARTER(date)

Function Description

Returns the number of quarters in the year represented by date, ranging from 1 to 4.

Example

Example 1: Returns the quarter in 2020 when "20-08-01" is used.

```
gbase> SELECT QUARTER('20-08-01') FROM dual;
+---------------------+
| QUARTER('20-08-01') |
+---------------------+
|                   3 |
+---------------------+
1 row in set
```

5.1.5.5.39 SECOND(time)

Function Description

Returns the number of seconds corresponding to time, ranging from 0 to 59.

Example

Example 1: Returns the number of seconds corresponding to "10:05:03".

```
gbase> SELECT SECOND('10:05:03') FROM dual;
+--------------------+
| SECOND('10:05:03') |
+--------------------+
|                  3 |
+--------------------+
1 row in set
```

5.1.5.5.40 SEC_ TO_ TIME(seconds)

Function Description

Returns the value of the parameter seconds in the format of "HH: MI: SS" or "HHMISS", which is converted to a value in hours, minutes, and seconds. The form of the returned value depends on whether the function is used in a string or numerical context.

Example

Example 1: After converting '2378' to hours, minutes, and seconds, return the value in the format of 'HH: MI: SS'.

```
gbase> SELECT SEC_ TO_ TIME(2378) FROM dual;
+-------------------+
| SEC_ TO_ TIME(2378) |
+-------------------+
| 00:39:38          |
+-------------------+
1 row in set
```

5.1.5.5.41 STR_ TO_ DATE(str,format)

Function Description

STR_ TO_ DATE() is DATE_ The inverse function of the FORMAT() function.

It obtains the string str and a formatted string format.

If the format string contains date and time parts, then STR_ TO_ DATE() returns a DATETIME, otherwise it returns a DATE or TIME value when only the date or time part is included.

The date, time, or datetime values included in the str should be given in the format. For the detailed format available, refer to the table in "5.1.5.5.13 DATE_FORMAT (date, FORMAT)".

If str contains an illegal date time or date time, STR_ TO_ DATE() returns NULL. An illegal value will also generate a warning.

Example

Example 1: The value of format is'% d.% m.% Y% H.% i '.

```
gbase> SELECT STR_ TO_ DATE('30.08.2020 09.20',
'%d.%m.%Y %H.%i') FROM dual;
+-----------------------------------------------------+
| STR_ TO_ DATE('30.08.2020 09.20', '%d.%m.%Y %H.%i') |
+-----------------------------------------------------+
| 2020-08-30 09:20:00                                 |
+-----------------------------------------------------+
1 row in set
```

Example 2: The value of format is'% Y -% m -% d% H:% i:% s'.

```
gbase> SELECT STR_ TO_ DATE('2020-15-08
00:00:00','%Y-%m-%d %H:%i:%s') FROM dual;
+-----------------------------------------------------+
```

```
| STR_ TO_ DATE('2020-15-08 00:00:00','%Y-%m-%d %H:%i:%s') |
+----------------------------------------------------------+
| NULL                                                     |
+----------------------------------------------------------+
1 row in set, 2 warnings
```

Example 3: The value of format is'% m/% d/% Y. '.

```
gbase> SELECT STR_ TO_ DATE('08/30/2020', '%m/%d/%Y') FROM dual;
+---------------------------------------+
| STR_ TO_ DATE('08/30/2020', '%m/%d/%Y') |
+---------------------------------------+
| 2020-08-30                            |
+---------------------------------------+
1 row in set
```

Example 4: Please note that the format '% X% V' cannot be used to convert a 'year week' string into a date because when a week crosses a month limit, a combination of year and week cannot indicate a unique year and month. If you want to convert 'year week' to a date, you should also specify a specific working day

```
gbase> SELECT str_ to_ date('202035 Monday', '%X%V %W') FROM dual;
+---------------------------------------+
| str_ to_ date('202035 Monday', '%X%V %W') |
+---------------------------------------+
| 2020-08-31                            |
+---------------------------------------+
1 row in set
```

5.1.5.5.42 SUBDATE()

grammar

SUBDATE(date,INTERVAL expr type)

SUBDATE(expr,days)

Function Description

When the second parameter called has INTERVAL, SUBDATe() is equivalent to DATE_ SUB()。 Please refer to "DATE_ADD(), DATE_SUB()" for specific information.

Expr is a date or datetime expression, with days used to subtract the number of days expr.

Example

Example 1: Using DATE_ SUB function, subtract 31 days from "2020 01 02".

```
gbase> SELECT DATE_ SUB('2020-01-02', INTERVAL 31 DAY) FROM
dual;
+-----------------------------------------+
| DATE_ SUB('2020-01-02', INTERVAL 31 DAY) |
+-----------------------------------------+
| 2019-12-02 00:00:00                     |
+-----------------------------------------+
1 row in set
```

Example 2: Using the SUBDATE function, subtract 31 days from "2020 01 02".

```
gbase> SELECT SUBDATE('2020-01-02', INTERVAL 31 DAY) FROM dual;
+---------------------------------------+
| SUBDATE('2020-01-02', INTERVAL 31 DAY) |
+---------------------------------------+
| 2019-12-02 00:00:00                   |
+---------------------------------------+
1 row in set
```

Example 3: Using the SUBDATE function, subtract 31 days from "2020 01 02 12:00:00".

```
gbase> SELECT SUBDATE('2020-01-02 12:00:00', 31) FROM dual;
+-------------------------------------+
| SUBDATE('2020-01-02 12:00:00', 31) |
+-------------------------------------+
| 2019-12-02 12:00:00                 |
+-------------------------------------+
1 row in set
```

5.1.5.5.43 SUBTIME(expr,expr2)

Function Description

SUBTIME() subtracts expr2 from expr and returns the result.

Expr is a time or datetime expression, and expr2 is a time expression.

Example

Example 1: Returns the value of "2020-12-31 23:59:59.999999" minus "1:1:1.000002".

```
gbase> SELECT SUBTIME('2020-12-31 23:59:59.999999','1 1:1:1.000002')
FROM dual;
+----------------------------------------------------------+
```

```
| SUBTIME('2020-12-31 23:59:59.999999','1 1:1:1.000002') |
+--------------------------------------------------------+
| 2020-12-30 22:58:58.999997                             |
+--------------------------------------------------------+
1 row in set
```

5.1.5.5.44 SYSDATE 、SYSDATE()

Function Description

Returns the current date time value in the format "YYYY-MM-DD HH: MI: SS" or "YYYYMMDDHHMISS", depending on whether the function is used in a string or numeric context.

Example

Example 1: Use the SYSDATE() function to return the current 'date+time'.

```
gbase> SELECT SYSDATE() FROM dual;
+---------------------+
| SYSDATE()           |
+---------------------+
| 2020-04-01 18:41:36 |
+---------------------+
1 row in set
```

5.1.5.5.45 TIME(expr)

Function Description

Extract the time part from the time or datetime expression expr and return it in string format.

Example

Example 1: Returns the time value from "2020 08 30 01:02:03" in string format.

```
gbase> SELECT TIME('2020-08-30 01:02:03') FROM dual;
+------------------------------+
| TIME('2020-08-30 01:02:03') |
+------------------------------+
| 01:02:03                     |
+------------------------------+
1 row in set
```

Example 2: Returns the time value in string format from "2020 08 30 01:02:03.000123".

```
gbase> SELECT TIME('2020-08-30 01:02:03.000123') FROM dual;
+------------------------------------+
| TIME('2020-08-30 01:02:03.000123') |
+------------------------------------+
| 01:02:03.000123                    |
+------------------------------------+
1 row in set
```

5.1.5.5.46 TIMEDIFF(expr,expr2)

Function Description

TIMEDIFF() returns the interval time between the start time expr and the end time expr2.

Expr and expr2 are time or datetime expressions, but both expressions must be of the same type.

Example

Example 1: Both expr and expr2 are time expressions that return the interval time between "2020:01:01 00:00:00:00" and "2020:01:01 00:00:0.00001".

```
gbase> SELECT TIMEDIFF('2020:01:01 00:00:00','2020:01:01
00:00:00.000001') FROM dual;
+-----------------------------------------------------------+
| TIMEDIFF('2020:01:01 00:00:00','2020:01:01 00:00:00.000001') |
+-----------------------------------------------------------+
| -00:00:00.000001                                          |
+-----------------------------------------------------------+
1 row in set
```

Example 2: Both expr and expr2 are datetime expressions that return the interval time between "2020 08 31 23:59:59.000001" and "2020 08 30 01:01:01.00002".

```
gbase> SELECT TIMEDIFF('2020-08-31 23:59:59.000001','2020-08-30
01:01:01.000002') FROM dual;
+-------------------------------------------------------------------+
| TIMEDIFF('2020-08-31 23:59:59.000001','2020-08-30 01:01:01.000002') |
+-------------------------------------------------------------------+
| 46:58:57.999999
|
+-------------------------------------------------------------------+
1 row in set
```

5.1.5.5.47 TIMESTAMP

grammar

TIMESTAMP(expr),TIMESTAMP(expr,expr2)

Function Description

TIMESTAMP (expr), returns the date or datetime expression expr based on the datetime value.

TIMESTAMP (expr, expr2), adds the time expression expr2 to the time expression expr, and returns a datetime value.

Example

Example 1: Using the TIMESTAMP (expr) function, return the datetime value corresponding to "2020 08 30".

```
gbase> SELECT TIMESTAMP('2020-08-30') FROM dual;
+-------------------------+
| TIMESTAMP('2020-08-30') |
+-------------------------+
| 2020-08-30 00:00:00     |
+-------------------------+
1 row in set
```

Example 2: Using the TIMESTAMP (expr, expr2) function, add "12:00:00" to "2020-12-31 12:00:00" and return its corresponding datetime value.

```
gbase> SELECT TIMESTAMP('2020-12-31 12:00:00','12:00:00') FROM
dual;
+-------------------------------------------+
| TIMESTAMP('2020-12-31 12:00:00','12:00:00') |
+-------------------------------------------+
| 2021-01-01 00:00:00                       |
+-------------------------------------------+
1 row in set
```

5.1.5.5.48 TIMESTAMPADD

grammar

TIMESTAMPADD(interval,int_expr,datetime_expr)

Function Description

Convert integer expression int_ Add expr to date or the expression datetime_ On expr.

int_ The unit of expr is given by the interval parameter and is one of the following values:

FRAC_ Second, Second, Minute, HOUR, DAY, WEEK, MONTH, QUARTER, or YEAR.

The interval value can be specified using the keywords above or using the prefix SQL_ TSI_。 For example, DAY or SQL_ TSI_ DAY, or both can be used.

Example

Example 1: Add "1" minute to "2020 01 02".

```
gbase> SELECT TIMESTAMPADD(MINUTE,1,'2020-01-02') FROM dual;
+--------------------------------------+
| TIMESTAMPADD(MINUTE,1,'2020-01-02') |
+--------------------------------------+
| 2020-01-02 00:01:00                  |
+--------------------------------------+
1 row in set
```

Example 2: Add '1' week to '2020 01 02'.

```
gbase> SELECT TIMESTAMPADD(WEEK,1,'2020-01-02') FROM dual;
+-----------------------------------+
| TIMESTAMPADD(WEEK,1,'2020-01-02') |
+-----------------------------------+
| 2020-01-09 00:00:00               |
+-----------------------------------+
1 row in set
```

5.1.5.5.49 TIMESTAMPDIFF

grammar

TIMESTAMPDIFF(interval,datetime_expr1,datetime_expr2)

Function Description

Returns date or the expression datetime as an integer_ Expr1 and datetime_ The difference between expr2 is given by the interval option.

The legal interval value is the same as the description of the TIMESTAMPADD() function.

Example

Example 1: Returns the number of months that differ between "2020 02 01" and "2020 05 01".

```
gbase> SELECT TIMESTAMPDIFF(MONTH,'2020-02-01','2020-05-01')
FROM dual;
+-----------------------------------------------+
| TIMESTAMPDIFF(MONTH,'2020-02-01','2020-05-01') |
+-----------------------------------------------+
|                                             3 |
+-----------------------------------------------+
1 row in set
```

Example 2: Returns the number of years that differ between "2020 05 01" and "2020 01 01 01".

```
gbase> SELECT TIMESTAMPDIFF(YEAR,'2020-05-01','2010-01-01')
FROM dual;
+-----------------------------------------------+
| TIMESTAMPDIFF(YEAR,'2020-05-01','2010-01-01') |
+-----------------------------------------------+
|                                           -10 |
+-----------------------------------------------+
1 row in set
```

5.1.5.5.50 TIME_ FORMAT(time,format)

Function Description

This function functions similarly to DATE_ The FORMAT() function, but the format string only supports formats such as hours, minutes, and seconds. Other incorrect formats may result in a NULL value or 0.

If the time value contains an hour portion greater than 23, the% H and% k hour formats will produce a value greater than the range of 0-23. The values generated in other hourly formats will be modeled using 12.

Example

Example 1: '100:00:00' contains a portion greater than 23 hours, '% H' and '% k' return '100'. The values generated by the formats'% h ','% I ', and'% l 'are' 100 MOD 12 '.

```
gbase> SELECT TIME_ FORMAT('100:00:00', '%H %k %h %I %l')
FROM dual;
+-------------------------------------------+
| TIME_ FORMAT('100:00:00', '%H %k %h %I %l') |
+-------------------------------------------+
```

```
| 100 100 04 04 4                                    |
+----------------------------------------------+
1 row in set
```

5.1.5.5.51 TIME_ TO_ SEC(time)

Function Description

Returns the number of seconds corresponding to the parameter time.

Example

Example 1: Returns the number of seconds corresponding to "22:23:00".

```
gbase> SELECT TIME_ TO_ SEC('22:23:00') FROM dual;
+-------------------------+
| TIME_ TO_ SEC('22:23:00') |
+-------------------------+
|                   80580 |
+-------------------------+
1 row in set
```

Example 2: Returns the number of seconds corresponding to "00:39:38".

```
gbase> SELECT TIME_ TO_ SEC('00:39:38') FROM dual;
+-------------------------+
| TIME_ TO_ SEC('00:39:38') |
+-------------------------+
|                    2378 |
+-------------------------+
1 row in set
```

5.1.5.5.52 TO_ DATE(string,format)

Function Description

Format the string string into a date of format type.

Table 525 Parameter Description

Format parameters	meaning
- ,	Usually used as a delimiter for parameter strings, regardless of the delimiter used, the formatted output format is always separated by "-". Because the date format of GBase 8a MPP Cluster is in ISO

Format parameters	meaning
. : / Space	format. Note: The string and FORMAT parameters must use consistent delimiters.
YYYY/YY	YYYY corresponds to 1-4 digits, and YY corresponds to 1-2 digits. Note: The value of YYYY or YY cannot be empty. If the FORMAT parameter does not specify YYYY or YY, it defaults to the current year, which is consistent with the year returned by NOW (). If both string and FORMAT are in the "YY-MM-DD" format, the result is returned, with the year being filled to 4 digits. The first two digits of the year are the first two digits of the value in the NOW () function. If the string is in the "YY-MM-DD" format and the format is in the "YYYY-MM-DD" format, the result is returned, and the year is also filled in 4 digits, but the first two digits of the year are filled in with "00". Year range: 0-9999.
MM	If the string parameter does not contain month MM, the format parameter defaults to returning the numerical value corresponding to the month in the NOW () function.
DD	If the string parameter does not contain daily DD, the format parameter defaults to 1 day.
DDD	The Nth day of the year. The parameter string must specify the year and days, and the FORMAT parameter should be in the form of YYYY DDD, so that the system can calculate the corresponding date.
HH、HH12、HH24	Format hours. If the hour part in the string parameter exceeds 12, then HH24 must be used for HH in the FORMAT parameter, otherwise an error will

Format parameters	meaning
	be reported.
AM	Short for morning, format the hours according to the 12 hour interval in the morning. Note: Do not use HH12 or HH24 formats, Both string and FORMAT parameters must specify both AM and PM. If the AM or PM specified by the string and FORMAT parameters are inconsistent, format the output using the string parameter to specify AM or PM.
PM	Short for afternoon, format the hours according to the 12 hour interval of the afternoon. Note: Do not use HH12 or HH24 formats, Both string and FORMAT parameters must specify both AM and PM. If the AM or PM specified by the string and FORMAT parameters are inconsistent, format the output using the string parameter to specify AM or PM.
MI	minute. If MI is omitted from the string and FORMAT parameters, the minute is returned as 0.
SS	Seconds. If SS is omitted in the string and FORMAT parameters, the return second is 0.

Example

Example 1: Regardless of the delimiter used for string and format, the formatting output is formatted with "-" as the delimiter.

```
gbase> SELECT TO_ DATE('2020-04-15','YYYY-MM-DD') FROM dual;
+----------------------------------+
```

```
| TO_ DATE('2020-04-15','YYYY-MM-DD') |
+-------------------------------------+
| 2020-04-15                          |
+-------------------------------------+
1 row in set

gbase> SELECT TO_ DATE('2020/04/15','YYYY/MM/DD') FROM dual;
+-------------------------------------+
| TO_ DATE('2020/04/15','YYYY/MM/DD') |
+-------------------------------------+
| 2020-04-15                          |
+-------------------------------------+
1 row in set

gbase> SELECT TO_ DATE('2020,04,15','YYYY,MM,DD') FROM dual;
+-------------------------------------+
| TO_ DATE('2020,04,15','YYYY,MM,DD') |
+-------------------------------------+
| 2020-04-15                          |
+-------------------------------------+
1 row in set

gbase> SELECT TO_ DATE('2020:4:15','YYYY:MM:DD') FROM dual;
+-------------------------------------+
| TO_ DATE('2020:4:15','YYYY:MM:DD') |
+-------------------------------------+
| 2020-04-15                          |
+-------------------------------------+
1 row in set

gbase> SELECT TO_ DATE('2020.4.15','YYYY.MM.DD') FROM dual;
+-------------------------------------+
| TO_ DATE('2020.4.15','YYYY.MM.DD') |
+-------------------------------------+
| 2020-04-15                          |
+-------------------------------------+
1 row in set

gbase> SELECT TO_ DATE('2020 4 14','YYYY MM DD') FROM dual;
+-------------------------------------+
| TO_ DATE('2020 4 14','YYYY MM DD') |
+-------------------------------------+
| 2020-04-14                          |
+-------------------------------------+
```

```
1 row in set
```

Example 2: If no year is specified in the string, the current year is returned, which is consistent with the year returned by NOW ().

```
gbase> SELECT TO_ DATE('4-15','MM-DD'),NOW() FROM dual;
+-------------------------+---------------------+
| TO_ DATE('4-15','MM-DD') | NOW()                |
+-------------------------+---------------------+
| 2020-04-15               | 2020-04-01 17:44:46 |
+-------------------------+---------------------+
1 row in set
```

If both string and format are in the "YY-MM-DD" format, the "year" in the returned result should be filled with 4 digits.

```
gbase> SELECT TO_ DATE('20-4-15','YY-MM-DD') FROM dual;
+-------------------------------+
| TO_ DATE('20-4-15','YY-MM-DD') |
+-------------------------------+
| 2020-04-15                    |
+-------------------------------+
1 row in set
```

If the string is in the "YY-MM-DD" format and the format is in the "YYYY-MM-DD" format, the year in the returned result is filled in 4 digits, but the first two digits of the year are filled in with "00".

```
gbase> SELECT TO_ DATE('20-04-15','YYYY-MM-DD') FROM dual;
+---------------------------------+
| TO_ DATE('20-04-15','YYYY-MM-DD') |
+---------------------------------+
| 0020-04-15                      |
+---------------------------------+
1 row in set
```

Example 3: If the string parameter does not contain month MM, the format parameter defaults to returning the numerical value corresponding to the month in the NOW () function.

```
gbase> SELECT TO_ DATE('2020-04-15','YYYY-MM-DD') FROM dual;
+-----------------------------------+
| TO_ DATE('2020-04-15','YYYY-MM-DD') |
+-----------------------------------+
| 2020-04-15                        |
+-----------------------------------+
1 row in set
```

```
gbase> SELECT TO_ DATE('2020-15','YYYY-DD') FROM dual;
+-------------------------------+
| TO_ DATE('2020-15','YYYY-DD') |
+-------------------------------+
| 2020-04-15                    |
+-------------------------------+
1 row in set
```

Example 4: If the string parameter does not contain daily DD, the format parameter defaults to returning 1.

```
gbase> SELECT TO_ DATE('2020-4-15','YYYY-MM-DD') FROM dual;
+-----------------------------------+
| TO_ DATE('2020-4-15','YYYY-MM-DD') |
+-----------------------------------+
| 2020-04-15                        |
+-----------------------------------+
1 row in set

gbase> SELECT TO_ DATE('2020-4','YYYY-MM') FROM dual;
+----------------------------+
| TO_ DATE('2020-4','YYYY-MM') |
+----------------------------+
| 2020-04-01                 |
+----------------------------+
1 row in set
```

Example 5: Returns the date corresponding to the nth day of the year.

```
gbase> SELECT TO_ DATE('2020    218','YYYY DDD') FROM dual;
+---------------------------------+
| TO_ DATE('2020    218','YYYY DDD') |
+---------------------------------+
| 2020-08-05                      |
+---------------------------------+
1 row in set

gbase> SELECT TO_ DATE('2020    55','YYYY DDD') FROM dual;
+---------------------------------+
| TO_ DATE('2020    55','YYYY DDD') |
+---------------------------------+
| 2020-02-24                      |
+---------------------------------+
1 row in set
```

```
gbase> SELECT TO_ DATE('2020 366','YYYY DDD') FROM dual;
+---------------------------------+
| TO_ DATE('2020 366','YYYY DDD') |
+---------------------------------+
| 2020-12-31                      |
+---------------------------------+
1 row in set
```

Example 6: Returns time in 12 hour and 24 hour formats.

```
gbase> SELECT TO_ DATE('2020-4-15 12:15:10','YYYY-MM-DD
HH:MI:SS') FROM dual;
+-----------------------------------------------------+
| TO_ DATE('2020-4-15 12:15:10','YYYY-MM-DD HH:MI:SS') |
+-----------------------------------------------------+
| 2020-04-15 12:15:10                                 |
+-----------------------------------------------------+
1 row in set

gbase> SELECT TO_ DATE('2020-4-15 12:15:10','YYYY-MM-DD
HH12:MI:SS') FROM dual;
+-----------------------------------------------------+
| TO_ DATE('2020-4-15 12:15:10','YYYY-MM-DD HH12:MI:SS') |
+-----------------------------------------------------+
| 2020-04-15 12:15:10                                 |
+-----------------------------------------------------+
1 row in set

gbase> SELECT TO_ DATE('2020-4-15 12:15:10','YYYY-MM-DD
HH24:MI:SS') FROM dual;
+-----------------------------------------------------+
| TO_ DATE('2020-4-15 12:15:10','YYYY-MM-DD HH24:MI:SS') |
+-----------------------------------------------------+
| 2020-04-15 12:15:10                                 |
+-----------------------------------------------------+
1 row in set
```

If the hourly part exceeds 12, HH24 must be used, otherwise an error will be reported.

```
gbase> SELECT TO_ DATE('2020-4-14 13:00:00','YYYY-MM-DD
HH:MI:SS') FROM dual;
ERROR 1708 (HY000): [172.168.83.12:5050](GBA-02AD-0005)Failed to query
in gnode:
DETAIL: incorrect parameter in function to_ date.
SQL: SELECT /*172.168.83.11_ 238_ 174_ 2020-04-01_ 18:54:11*/ /*+
TID('196780') */ to_ date('2020-4-14 13:00:00', 'YYYY-MM-DD HH:MI:SS') AS
```

```
`to_ date('2020-4-14 13:00:00','yyyy-mm-dd hh:mi:ss')` FROM `gclusterdb`.` dual`
`vcname000001.gclusterdb.dual`
```

After using HH24, if the hour portion exceeds 12, the output is formatted smoothly.

```
gbase> SELECT TO_ DATE('2020-4-14 13:00:00','YYYY-MM-DD
HH24:MI:SS') FROM dual;
+-----------------------------------------------------+
| TO_ DATE('2020-4-14 13:00:00','YYYY-MM-DD HH24:MI:SS') |
+-----------------------------------------------------+
| 2020-04-14 13:00:00                                 |
+-----------------------------------------------------+
1 row in set
```

The following example is used to illustrate the TO of GBase 8a MPP Cluster_ The DATE() function is flexible.

```
gbase> SELECT TO_ DATE('2020 1111','YYYY MISS') FROM dual;
+---------------------------------+
| TO_ DATE('2020 1111','YYYY MISS') |
+---------------------------------+
| 2020-04-01 00:11:11             |
+---------------------------------+
1 row in set
```

Example 7: Both string and format parameters specify AM or PM simultaneously.

```
gbase> SELECT TO_ DATE('2020-4-15 8:15:20 AM','YYYY-MM-DD
HH:MI:SS AM') FROM dual;
+-----------------------------------------------------+
| TO_ DATE('2020-4-15 8:15:20 AM','YYYY-MM-DD HH:MI:SS AM') |
+-----------------------------------------------------+
| 2020-04-15 08:15:20                                 |
+-----------------------------------------------------+
1 row in set

gbase> SELECT TO_ DATE('2020-4-15 8:15:20 PM','YYYY-MM-DD
HH:MI:SS PM') FROM dual;
+-----------------------------------------------------+
| TO_ DATE('2020-4-15 8:15:20 PM','YYYY-MM-DD HH:MI:SS PM') |
+-----------------------------------------------------+
| 2020-04-15 20:15:20                                 |
+-----------------------------------------------------+
1 row in set
```

If the AM or PM specified by the string and format parameters are inconsistent, use the string parameter to specify AM or PM for formatting output.

```
gbase> SELECT TO_ DATE('2020-4-15 8:15:20 AM','YYYY-MM-DD
HH:MI:SS PM') FROM dual;
+-----------------------------------------------------------+
| TO_ DATE('2020-4-15 8:15:20 AM','YYYY-MM-DD HH:MI:SS PM') |
+-----------------------------------------------------------+
| 2020-04-15 08:15:20                                       |
+-----------------------------------------------------------+
1 row in set

gbase> SELECT TO_ DATE('2020-4-15 8:15:20PM','YYYY-MM-DD
HH:MI:SS AM') FROM dual;
+-----------------------------------------------------------+
| TO_ DATE('2020-4-15 8:15:20PM','YYYY-MM-DD HH:MI:SS AM') |
+-----------------------------------------------------------+
| 2020-04-15 20:15:20                                       |
+-----------------------------------------------------------+
1 row in set
```

Example 8: Comparing NOW () and TO_ DATE(TO_CHAR(NOW(),'YYYY-MM-DD HH:MI:SS'),'YYYY-MM-DD HH:MI:SS')。

```
gbase> SELECT NOW(),TO_ DATE(TO_CHAR(NOW(),'YYYY-MM-DD
HH:MI:SS'),'YYYY-MM-DD HH:MI:SS') AS f_ format FROM dual;
+---------------------+---------------------+
| NOW()               | f_ format           |
+---------------------+---------------------+
| 2020-04-01 18:57:18 | 2020-04-01 06:57:18 |
+---------------------+---------------------+
1 row in set
```

TO_ CHAR() omits minutes.

```
gbase> SELECT NOW(),TO_ DATE(TO_CHAR(NOW(),'YYYY-MM-DD
HH:SS'),'YYYY-MM-DD HH:SS') AS f_ format FROM dual;
+---------------------+---------------------+
| NOW()               | f_ format           |
+---------------------+---------------------+
| 2020-04-01 18:57:34 | 2020-04-01 06:00:34 |
+---------------------+---------------------+
1 row in set
```

Example 9: Comparing NOW () and TO_ DATE(TO_CHAR(NOW(),'YYYY-MM-DD HH:MI:SS'),'YYYY-MM-DD HH:MI:SS')。

```
gbase> SELECT NOW(),TO_ DATE(TO_CHAR(NOW(),'YYYY-MM-DD
HH:MI:SS'),'YYYY-MM-DD HH:MI:SS') AS f_ format FROM dual;
+---------------------+---------------------+
```

```
| NOW()                 | f_ format           |
+---------------------+---------------------+
| 2020-04-01 18:57:46 | 2020-04-01 06:57:46 |
+---------------------+---------------------+
1 row in set
```

TO_ CHAR() omits seconds.

```
gbase> SELECT NOW(),TO_ DATE(TO_CHAR(NOW(),'YYYY-MM-DD
HH:MI'),'YYYY-MM-DD HH:MI') AS f_ format FROM dual;
+---------------------+---------------------+
| NOW()                 | f_ format           |
+---------------------+---------------------+
| 2020-04-01 18:57:57 | 2020-04-01 06:57:00 |
+---------------------+---------------------+
1 row in set
```

5.1.5.5.53 TO_ DAYS(date)

Function Description

Returns the number of days corresponding to the date (starting from year 0)

Example

Example 1: Returns the number of days corresponding to "990501".

```
gbase> SELECT TO_ DAYS(990501) FROM dual;
+-----------------+
| TO_ DAYS(990501) |
+-----------------+
|          730240 |
+-----------------+
1 row in set
```

Example 2: Returns the number of days corresponding to "2020 08 30".

```
gbase> SELECT TO_ DAYS('2020-08-30') FROM dual;
+-----------------------+
| TO_ DAYS('2020-08-30') |
+-----------------------+
|                738032 |
+-----------------------+
1 row in set
```

TO_ DAYS() is not used for values before the Gregorian calendar (1582) because missing dates are not taken into account when the calendar changes.

GBase 8a MPP Cluster uses rules in date and time types to convert year values from two dates to four digits.

Example 3: "2020 08 30" and "20 08 30" represent the same date.

```
gbase> SELECT TO_ DAYS('2020-08-30'), TO_ DAYS('20-08-30') FROM dual;
+----------------------+---------------------+
| TO_ DAYS('2020-08-30') | TO_ DAYS('20-08-30') |
+----------------------+---------------------+
|                 738032 |                738032 |
+----------------------+---------------------+
1 row in set
```

Example 4: For dates before 1582 (perhaps the next year in other regions), the results are unreliable.

```
gbase> SELECT TO_ DAYS('1581-08-30') FROM dual;
+-----------------------+
| TO_ DAYS('1581-08-30') |
+-----------------------+
|                577690 |
+-----------------------+
1 row in set
```

5.1.5.5.54 TRUNC(date/datetime[, format])

Function Description

The TRUNC function returns a date value that is truncated in the specified element format.

Table 526 Parameter Description

Parameter Name	Description
date/datetime	It is a required parameter that represents a date value entered.
format	Optional parameter, representing the date format, used to truncate the input date value in the specified element format. If this parameter is omitted, it will be truncated by the most recent date.

Table 527 Format Support Type Description

Type	explain
year	Return to the first day of the year
yyyy	Return to the first day of the year
month	Return to the first day of the month
mm	Return to the first day of the month

Type	explain
dd	Return the date of the day
hh	Obtain the date of 0:00:00:00 on the same day
mi	Obtain the date of 0:00:00:00 on the same day

Example

Example 1: Returns the first day of the year of execution.

```
gbase> SELECT TRUNC(current_date,'year') FROM dual;
+----------------------------+
| TRUNC(current_date,'year') |
+----------------------------+
| 2020-01-01                 |
+----------------------------+
1 row in set
```

Example 2: Returns the first day of the year of execution.

```
gbase> SELECT TRUNC(current_date,'yyyy') FROM dual;
+----------------------------+
| TRUNC(current_date,'yyyy') |
+----------------------------+
| 2020-01-01                 |
+----------------------------+
1 row in set
```

Example 3: Returns the first day of the month of execution.

```
gbase> SELECT TRUNC(current_date,'mm') FROM dual;
+--------------------------+
| TRUNC(current_date,'mm') |
+--------------------------+
| 2020-04-01               |
+--------------------------+
1 row in set
```

Example 4: Returns the date of the execution day.

```
gbase> SELECT TRUNC(current_date,'dd') FROM dual;
+--------------------------+
| TRUNC(current_date,'dd') |
+--------------------------+
| 2020-04-02               |
+--------------------------+
1 row in set
```

Example 5: Obtain the date of the day at 00:00:00 am.

```
gbase> SELECT TRUNC(current_date,'hh') FROM dual;
+--------------------------+
| TRUNC(current_date,'hh') |
+--------------------------+
| 2020-04-02 00:00:00      |
+--------------------------+
1 row in set
```

Example 6: Obtain the date of the day at 00:00:00 am.

```
gbase> SELECT TRUNC(current_date,'mi') FROM dual;
+--------------------------+
| TRUNC(current_date,'mi') |
+--------------------------+
| 2020-04-02 00:00:00      |
+--------------------------+
1 row in set
```

5.1.5.5.55 UNIX_ TIMESTAMP

grammar

UNIX_ TIMESTAMP(), UNIX_ TIMESTAMP(date)

Function Description

If there are no parameters when calling, return a Unix timestamp in unsigned integer form (seconds starting from "1970 01 01 00:00:00" GMT).

If calling UNIX with a parameter date_ TIMESTAMP(), which returns the number of seconds that the parameter value has passed since "1970-01-01 00:00:00" GMT.

Date can be a DATE string, a DATETIME string, a TIMESTAMP, displayed as a local time in either YYMMDD or YYYYMMDD.

Example

Example 1: UNIX_ TIMESTAMP() has no parameters and returns a Unix timestamp as an unsigned integer.

```
gbase> SELECT UNIX_ TIMESTAMP() FROM dual;
+------------------+
| UNIX_ TIMESTAMP() |
+------------------+
|       1585740352 |
+------------------+
1 row in set
```

Example 2: UNIX_ TIMESTAMP (date), returns the number of seconds passed from "1970 01 01 00:00:00" GMT to "2020 04 10 22:23:00".

```
gbase> SELECT UNIX_ TIMESTAMP('2020-04-10 22:23:00') FROM dual;
+----------------------------------------+
| UNIX_ TIMESTAMP('2020-04-10 22:23:00') |
+----------------------------------------+
|                                 1586528580 |
+----------------------------------------+
1 row in set
```

When UNIX_ When TIMESTAMP is used in the TIMESTAMP column, the function directly returns the internal timestamp value without any implicit "string to Unix timestamp" conversion. If you want to move to UNIX_ TIMESTAMP() passes an overflow date, which returns 0, but please note that only basic range checking is performed (year from 1970 to 2037, month from 01 to 12, date from 01 to 31).

be careful

- If the user wishes to subtract UNIX_ TIMESTAMP() column, the result needs to be cast to a signed integer.

5.1.5.5.56 UTC_ DATE, UTC_ DATE()

Function Description

Returns the current UTC date in the format of "YYYY-MM-DD" or "YYYYMMDD", depending on whether the function is used in a string or numerical context.

Example

Example 1: Returns the current UTC date in the format of "YYYY-MM-DD" or "YYYYMMDD".

```
gbase> SELECT UTC_ DATE(), UTC_ DATE() + 0 FROM dual;
+------------+----------------+
| UTC_ DATE() | UTC_ DATE() + 0 |
+------------+----------------+
| 2020-04-01 | 2020-04-01          |
+------------+----------------+
1 row in set
```

5.1.5.5.57 UTC_ TIME，UTC_ TIME()

Function Description

Returns the current UTC time in the format of "HH: MI: SS" or "HHMISS", depending on whether the function is used in a string or numerical context.

Example

Example 1: Returns the current UTC time in the format "HH: MI: SS".

```
gbase> SELECT UTC_ TIME() FROM dual;
+------------+
| UTC_ TIME() |
+------------+
| 11:29:11      |
+------------+
1 row in set
```

5.1.5.5.58 UTC_ TIMESTAMP，UTC_ TIMESTAMP()

Function Description

Returns the current UTC date in the format "YYYY-MM-DD HH: MI: SS" or "YYYYMMDDHHMISS", depending on whether the function is used in a string or numerical context.

Example

Example 1: Returns the current UTC "date+time" in the format of "YYYY-MM-DD HH: MI: SS" or YYYYMMDDHHMISS.

```
gbase> SELECT UTC_ TIMESTAMP(), UTC_ TIMESTAMP() + 0 FROM
dual;
+---------------------+---------------------+
| UTC_ TIMESTAMP()          | UTC_ TIMESTAMP() + 0 |
+---------------------+---------------------+
| 2020-04-01 11:29:29 | 2020-04-01 11:29:29 |
+---------------------+---------------------+
1 row in set
```

5.1.5.5.59 WEEK(date[,mode])

Function Description

This function returns the number of weeks of the date.

The two parameter form WEEK() allows the user to specify whether the week starts on a Sunday or Monday, as well as the return value range.

If the mode parameter is ignored, the system variable default is used_ week_ The value of format.

explain

SHOW VARIABLES LIKE '% default' can be used_ week_ format%'; Command View Default_ week_ The default value of format.

The following table shows how the mode parameter works:

Table 528 Parameter Description

pattern	Starting day of the week	Range	explain
0	Sunday	0~53	The first week starts on Sunday
one	Monday	0~53	This mode has been used for an additional 3 days
two	Sunday	1~53	The first week starts on Sunday
three	Monday	1~53	This mode has been used for an additional 3 days
four	Sunday	0~53	This mode has been used for an additional 3 days
five	Monday	0~53	The first week starts on Monday
six	Sunday	1~53	This mode has been used for an additional 3 days
seven	Monday	1~53	The first week starts on Monday
eight	Sunday	1~54	Sunday is the beginning of the week, and any day is counted as a week.
nine	Monday	1~54	Monday is the beginning of the week, and as long as there is one day, it is considered a week.

Example

Example 1: Returns the week in 2020 corresponding to "2020 08 31".

```
gbase> SELECT WEEK('2020-08-31') FROM dual;
+--------------------+
| WEEK('2020-08-31') |
+--------------------+
```

```
|                          35 |
+--------------------+
1 row in set
```

Example 2: Returns the week in 2020 corresponding to "2020 08 30", with a mode of "0".

```
gbase> SELECT WEEK('2020-08-20',0) FROM dual;
+----------------------+
| WEEK('2020-08-20',0) |
+----------------------+
|                   33 |
+----------------------+
1 row in set
```

Example 3: Returns the week in 2020 corresponding to "2020 08 30", with the mode "1".

```
gbase> SELECT WEEK('2020-08-30',1) FROM dual;
+----------------------+
| WEEK('2020-08-30',1) |
+----------------------+
|                   35 |
+----------------------+
1 row in set
```

Example 4: Returns the week ordinal of 2020 corresponding to "2020-12-31", with mode "1".

```
gbase> SELECT WEEK('2020-12-31',1) FROM dual;
+----------------------+
| WEEK('2020-12-31',1) |
+----------------------+
|                   53 |
+----------------------+
1 row in set
```

Example 5: If a week is the last week of the previous year and optional modes 2, 3, 6, or 7 are not used, the function WEEK() returns 0.

```
gbase> SELECT YEAR('2020-01-01'), WEEK('2020-01-01',0) FROM dual;
+--------------------+----------------------+
| YEAR('2020-01-01') | WEEK('2020-01-01',0) |
+--------------------+----------------------+
|               2020 |                    0 |
+--------------------+----------------------+
1 row in set
```

Example 6: Returns the week in 2020 corresponding to "2020 01 01", with the mode "2".

```
gbase> SELECT YEAR('2020-01-01'), WEEK('2020-01-01',2) FROM dual;
+--------------------+----------------------+
| YEAR('2020-01-01') | WEEK('2020-01-01',2) |
+--------------------+----------------------+
|               2020 |                   52 |
+--------------------+----------------------+
1 row in set
```

Example 7: Returns the week in 2020 corresponding to "2020 01 01", with the mode "3".

```
gbase> SELECT YEAR('2020-01-01'), WEEK('2020-01-01',3) FROM dual;
+--------------------+----------------------+
| YEAR('2020-01-01') | WEEK('2020-01-01',3) |
+--------------------+----------------------+
|               2020 |                    1 |
+--------------------+----------------------+
1 row in set
```

Example 8: Returns the week in 2020 corresponding to "2020 01 01", with the mode "6".

```
gbase> SELECT YEAR('2020-01-01'), WEEK('2020-01-01',6) FROM dual;
+--------------------+----------------------+
| YEAR('2020-01-01') | WEEK('2020-01-01',6) |
+--------------------+----------------------+
|               2020 |                    1 |
+--------------------+----------------------+
1 row in set
```

Example 9: Returns the week in 2020 corresponding to "2020 01 01", with a mode of "0".

```
gbase> SELECT YEAR('2020-01-01'), WEEK('2020-01-01',0) FROM dual;
+--------------------+----------------------+
| YEAR('2020-01-01') | WEEK('2020-01-01',0) |
+--------------------+----------------------+
|               2020 |                    0 |
+--------------------+----------------------+
1 row in set
```

Users may provide feedback that GBase 8a MPP Cluster should return "52" for the WEEK() function, as the given date actually occurs in week 52 of 2019. The reason we decided to return '0' as a replacement is because we hope that the function can return 'the number of weeks in a given year'. This makes the WEEK() function more reliable when combined with other functions that extract date parts from dates.

If you want the calculated results for the year to include the first day of the week where the given date is located, you should use 0, 2, 5, or 7 as the mode parameter selection.

Example 10: Returns the week in 2020 corresponding to "2020 01 01", with the mode "2".

```
gbase> SELECT WEEK('2020-01-01',2) FROM dual;
+----------------------+
| WEEK('2020-01-01',2) |
+----------------------+
|                   52 |
+----------------------+
1 row in set
```

Example 11: Using the YEARWEEK() function.

```
gbase> SELECT YEARWEEK('2020-01-01') FROM dual;
+------------------------+
| YEARWEEK('2020-01-01') |
+------------------------+
|                 201952 |
+------------------------+
1 row in set
```

Example 12: Using the MID function, extract 2 consecutive characters from the return value of "YEARWEEK ('2020 01 01 ')" starting from the first character.

```
gbase> SELECT MID(YEARWEEK('2020-01-01'),5,2) FROM dual;
+-----------------------------------+
| MID(YEARWEEK('2020-01-01'),5,2) |
+-----------------------------------+
| 52                                |
+-----------------------------------+
1 row in set
```

5.1.5.5.60 WEEKDAY(date)

Function Description

Returns the week index corresponding to date (0=Monday, 1=Tuesday,... 6=Sunday).

Example

Example 1: Returning "2020 08 29 22:23:00" corresponds to the day of the week.

```
gbase> SELECT WEEKDAY('2020-08-29 22:23:00') FROM dual;
+--------------------------------+
| WEEKDAY('2020-08-29 22:23:00') |
+--------------------------------+
```

```
|                                    5 |
+--------------------------------+
1 row in set
```

Example 2: Returning "2020 08 05" corresponds to the day of the week.

```
gbase> SELECT WEEKDAY('2020-08-05') FROM dual;
+-----------------------+
| WEEKDAY('2020-08-05') |
+-----------------------+
|                     2 |
+-----------------------+
1 row in set
```

5.1.5.5.61 WEEKOFYEAR(date)

Function Description

Returns the number of weeks for date.

Table 529 Parameter Description

Parameter Name	Description
date	The value range is 1-53.
WEEKOFYEAR(date)	Equivalent to WEEK (date, 3).

Example

Example 1: Returning "2020 08 30" corresponds to the week of 2020.

```
gbase> SELECT WEEKOFYEAR('2020-08-30') FROM dual;
+-------------------------+
| WEEKOFYEAR('2020-08-30') |
+-------------------------+
|                      35 |
+-------------------------+
1 row in set
```

5.1.5.5.62 YEAR(date)

Function Description

Returns the year value corresponding to the date.

Example

Example 1: Returns the year corresponding to "2020 02 03".

```
gbase> SELECT YEAR('2020-02-03') FROM dual;
+--------------------+
| YEAR('2020-02-03') |
+--------------------+
|                2020 |
+--------------------+
1 row in set
```

5.1.5.5.63 YEARWEEK(date), YEARWEEK(date,mode)

Function Description

Returns the year and week corresponding to the date date.

The parameter mode in YEARWEEK (date, mode) has the same form and effect as the mode parameter in WEEK(). Similarly, if the mode parameter is ignored, the system variable default is used_ week_ The value of format.

Example

Example 1: Returns the year and week corresponding to "2020 02 01", with a mode of 0.

```
gbase> SELECT YEARWEEK('2020-02-01',0) FROM dual;
+--------------------------+
| YEARWEEK('2020-02-01',0) |
+--------------------------+
|                   202004 |
+--------------------------+
1 row in set
```

explain

The number of weeks in the return value of the Yearweek function is not allowed to be 0. When the number of weeks is 0, the last week of the previous year will be returned, and the corresponding year in the return value must also be modified accordingly. This is different from the week function.

Example 2: When the date parameter date is the first or last week of a year, the returned year value may not match the year given by the date parameter.

```
gbase> SELECT YEARWEEK('2020-01-01') FROM dual;
+------------------------+
```

```
| YEARWEEK('2020-01-01') |
+------------------------+
|                 201952 |
+------------------------+
1 row in set
```

When the parameter start of the YEARWEEK() function has a value of 0 or 1, the return value of the week value is different from the return value (0) of the WEEK() function, which returns the week value based on the given year.

```
gbase> SELECT YEARWEEK('2020-01-01',0),WEEK('2020-01-01',0)
FROM dual;
+-------------------------+----------------------+
| YEARWEEK('2020-01-01',0) | WEEK('2020-01-01',0) |
+-------------------------+----------------------+
|                  201952 |                    0 |
+-------------------------+----------------------+
1 row in set

gbase> SELECT YEARWEEK('2020-01-01',1),WEEK('2020-01-01',1) FROM
dual;
+-------------------------+----------------------+
| YEARWEEK('2020-01-01',1) | WEEK('2020-01-01',1) |
+-------------------------+----------------------+
|                  202001 |                    1 |
+-------------------------+----------------------+
1 row in set
```

5.1.5.6 Other functions

5.1.5.6.1 **Bit function**

Function Description

GBase 8a MPP Cluster uses the BIGINT (64 bit) algorithm for bit operations, so the maximum effective range of these operators is 64 bits.

be careful

Bitwise function operations only support numerical types.

5.1.5.6.1.1 |按位或

Example

Returns the calculation result of '29 | 15'.

```
gbase> SELECT 29 | 15 FROM dual;
+---------+
| 29 | 15 |
+---------+
|          31 |
+---------+
1 row in set
```

explain

The bit value corresponding to 29 is "11101", and the bit value corresponding to 15 is "1111". When performing the OR operation bit by bit, the result is "11111", and the corresponding decimal value is "31".

5.1.5.6.1.2 &Bitwise and

Example

Returns the calculation result of '29&15'.

```
gbase> SELECT 29 & 15 FROM dual;
+---------+
| 29 & 15 |
+---------+
|          13 |
+---------+
1 row in set
```

explain

Example explanation: The bit value corresponding to "29" is "11101", and the bit value corresponding to "15" is "1111". By performing a bitwise AND operation, the result is "1101", and the corresponding decimal value is "13".

5.1.5.6.1.3 ^Bitwise XOR

Example

Example 1: Returns the calculation result of "1 ^ 1".

```
gbase> SELECT 1 ^ 1 FROM dual;
+-------+
| 1 ^ 1 |
+-------+
|     0 |
+-------+
1 row in set
```

Example 2: Returns the calculation result of "1 ^ 0".

```
gbase> SELECT 1 ^ 0 FROM dual;
+-------+
| 1 ^ 0 |
+-------+
|     1 |
+-------+
1 row in set
```

Example 3: Returns the calculation result of "11 ^ 3".

```
gbase> SELECT 11 ^ 3 FROM dual;
+--------+
| 11 ^ 3 |
+--------+
|      8 |
+--------+
1 row in set
```

explain

The bit value corresponding to "11" is "1011", and the bit value corresponding to "3" is "0011". XOR is performed bit by bit, resulting in "1000" and the corresponding decimal value is "8".

5.1.5.6.1.4 <<Shift Left Operation (BIGINT)

Example

Example 1: Returns the calculation result of "1<<2".

```
gbase> SELECT 1 << 2 FROM dual;
+--------+
| 1 << 2 |
+--------+
|      4 |
+--------+
1 row in set
```

 **explain**

The bit value corresponding to '1' is' 0001 ', the left shift of two bits is' 0100', and the corresponding decimal is' 1 '.

5.1.5.6.1.5 >>Move Right Operation (BIGINT)

Example

Returns the calculation result of "4>>2".

```
gbase> SELECT 4 >> 2 FROM dual;
+--------+
| 4 >> 2 |
+--------+
|      1 |
+--------+
1 row in set
```

 **explain**

Example explanation: The bit value corresponding to "4" is "0100", the right shift of two bits is "0001", and the corresponding decimal value is "1".

5.1.5.6.1.6 BIT_ COUNT(N)

Function Description

Returns the total number of bits set in parameter N that are 1.

Example

Returns the number of 1 in the bit set by "29".

```
gbase> SELECT BIT_ COUNT(29) FROM dual;
```

```
+---------------+
| BIT_ COUNT(29) |
+---------------+
|                4 |
+---------------+
1 row in set
```

explain

The bit value corresponding to "29" is "11101", and the number of 1 in the corresponding bit is "4".

5.1.5.6.2 Encryption function

Function Description

This part of the function is used to encrypt and decrypt data.

5.1.5.6.2.1 AES_ ENCRYPT

Function Description

AES_ ENCRYPT(str,key_str)

This function allows the use of the official AES algorithm to encrypt data, formerly known as "Rijndael". The encoding uses a key length of 128 bits. The input parameter can be of any length. If the parameter is NULL, the return result of the function is also NULL; If AES_ DECRYPT() detects invalid data or incorrect padding and returns NULL. AES_ ENCRYPT() is currently the most encryption secure function in GBase 8a MPP Cluster.

5.1.5.6.2.2 AES_ DECRYPT

Function Description

AES_ DECRYPT is an AES_ The decryption function for DECRYPT().

Example

```
gbase> set @pass=AES_ ENCRYPT('hello,world','key');
Query OK, 0 rows affected (Elapsed: 00:00:00.09)
```

```
gbase> select char_ length(@pass);
+--------------------+
| char_ length(@pass) |
+--------------------+
|                  16 |
+--------------------+
1 row in set (Elapsed: 00:00:00.00)
gbase> select AES_ DECRYPT(@pass,'key');
+--------------------------+
| AES_ DECRYPT(@pass,'key') |
+--------------------------+
| hello,world              |
+--------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.6.2.3 ENCRYPT(str[,salt])

Function Description

Use Linux's crypt() system call to encrypt str. The parameter salt is a string that contains at least two characters. If salt is not given, a random value will be used.

Example

Since no salt value was given, a random value was used to encrypt 'hello'.

```
gbase> SELECT ENCRYPT('hello') FROM dual;
+------------------+
| ENCRYPT('hello') |
+------------------+
| y/oLk8SmVyZXg    |
+------------------+
1 row in set
```

explain

ENCRYPT() ignores all characters except for the first 8 characters of str on some systems, and this behavior is determined by using the crypt() system call.

If crypt() is not available on the user's system, ENCRYPT() always returns NULL, so it is recommended that users use MD5() or SHA1(), which exist on all platforms.

5.1.5.6.2.4 MD5(str)

Function Description

Calculate a 128-bit MD5 checksum for a string, and return the result as a 32-bit hexadecimal string. The return value can be used as a hash key. If the parameter is NULL, it returns NULL.

Example

Use MD5() to encrypt 'testing'.

```
gbase> SELECT MD5('testing') FROM dual;
+----------------------------------+
| MD5('testing')                   |
+----------------------------------+
| ae2b1fca515949e5d54fb22b8ed95575 |
+----------------------------------+
1 row in set
```

explain

This is RSA Data Security, Inc. MD5 Message Digest Algorithm. If the user wants to convert the value to uppercase, refer to the BINARY operator described in the conversion functions and operators, which can be used for binary string conversion.

5.1.5.6.2.5 SHA1(str), SHA(str)

Function Description

Calculate a 160 bit SHA1 checksum for a string as described in RFC3174 (Secure Hash Algorithm), and return the result as a 40 bit hexadecimal string; If the value of str is NULL, it returns NULL. Commonly used as a hash key. Users can also use it as an encryption security function to store passwords.

Example

Calculate a 160 bit SHA1 checksum for 'abc' and return the result as a 40 bit hexadecimal string.

```
gbase> SELECT SHA1('abc') FROM dual;
+------------------------------------------+
| SHA1('abc')                              |
```

```
+------------------------------------------+
| a9993e364706816aba3e25717850c26c9cd0d89d |
+------------------------------------------+
1 row in set
```

explain

SHA1() can be considered as equivalent to MD5() in terms of encryption security, and SHA() is a synonym for SHA1().

5.1.5.6.2.6 to_ base64(str)

Function Description

to_ base64(str)

Implement base64 encoding encryption for data.

The maximum length allowed for parameter str is 12419496 (bytes), and an error is reported if it is too long.

The length of the execution result of this function is influenced by max_ allowed_ Packet restriction, excessive length error reported.

```
gbase> select to_ base64('hello');
+--------------------+
| to_ base64('hello') |
+--------------------+
| aGVsbG8=           |
+--------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.6.2.7 from_ base64(str)

Function Description

from_ base64(str)

Implement base64 decoding of data.

The maximum length allowed for parameter str is 16M, and an error is reported if it is too long.

The length of the execution result of this function is influenced by max_ allowed_ Packet restriction, excessive length error reported.

```
gbase> select from_ base64('aGVsbG8=');
+-------------------------+
| from_ base64('aGVsbG8=') |
+-------------------------+
| hello                   |
+-------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.6.2.8 SHA2(str)

Function Description

The sha2 function encrypts the input string, generates a fixed length string, and outputs it.

The SHA2 (str string, mode int) function selects the corresponding SHA-2 algorithm based on the mode to encrypt the string str.

The SHA-224, SHA-256, SHA-384, and SHA-512 algorithms can encrypt strings into fixed length hexadecimal strings, and these four algorithms are collectively referred to as the SHA-2 algorithm.

The newly added SHA2 function can implement the above four encryption methods.

grammar

```
SHA2(str string,mode int)
```

Table -597 Parameter Description

Parameter Name	explain
str	String, which can contain Chinese, English, numbers, as well as symbols, special symbols, mathematical notation and other symbols in the ascii table.
mode	An int type number used for mode selection, optional as 224256384512. If it is not within the above range, it will output null.

explain

A series of tools are provided in the existing file<openssl/sha. h>in the Extra directory to implement the SHA-2 algorithm, which can be directly called.

Example

```
gbase> select sha2('123abc',224);
+----------------------------------------------------------+
| sha2('123abc',224)                                       |
+----------------------------------------------------------+
| 867916c509e025cd1eef93151e82f518fcfdfcacd759965fd99e27af |
+----------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.08)

gbase> select sha2('123abc',256);
+------------------------------------------------------------------+
| sha2('123abc',256)                                               |
+------------------------------------------------------------------+
| dd130a849d7b29e5541b05d2f7f86a4acd4f1ec598c1c9438783f56bc4f0ff80 |
+------------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.02)


gbase> select sha2('123abc',384);
+--------------------------------------------------------------------------------------------------+
|sha2('123abc',384)                                                                                |
+--------------------------------------------------------------------------------------------------+
|40889bd7aee5be35f9d713c4f51ff00fe2099fd901d094284d6edc9aa39ec096cebe9547d3cbaf1f35101a
6489f033c3                                                                                         |
+--------------------------------------------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.03)


gbase> select sha2('123abc',512);
+----------------------------------------------------------------------------------------------------------------------------------+
|sha2('123abc',512)
|
+----------------------------------------------------------------------------------------------------------------------------------+
|7b6ad79b346fb6951275343948e13c1b4ebca82a5452a6c5d15684377f096ca927506a23a847e6e04606
1399631b16fc2820c8b0e02d0ea87aa5a203a77c2a7e
|
+----------------------------------------------------------------------------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> select sha2('123abc',111);
+--------------------+
| sha2('123abc',111) |
+--------------------+
```

```
| NULL                    |
+-------------------+
1 row in set, 1 warning (Elapsed: 00:00:00.00)
```

Four algorithms are used to encrypt strings into fixed length strings of different lengths.

```
gbase> select length(sha224)from (select sha2(str,224)as sha224 from t)as tmp_ table;
+----------------+
| length(sha224) |
+----------------+
|                56 |
|                56 |
+----------------+
2 rows in set (Elapsed: 00:00:00.17)

gbase> select length(sha256)from (select sha2(str,256)as sha256 from t)as tmp_ table;
+----------------+
| length(sha256) |
+----------------+
|                64 |
|                64 |
+----------------+
2 rows in set (Elapsed: 00:00:00.09)

gbase> select length(sha384)from (select sha2(str,384)as sha384 from t)as tmp_ table;
+----------------+
| length(sha384) |
+----------------+
|                96 |
|                96 |
+----------------+
2 rows in set (Elapsed: 00:00:00.05)

gbase> select length(sha512)from (select sha2(str,512)as sha512 from t)as tmp_ table;
+----------------+
| length(sha512) |
+----------------+
|               128 |
|               128 |
+----------------+
2 rows in set (Elapsed: 00:00:00.02)
```

5.1.5.6.3 Information function

5.1.5.6.3.1 BENCHMARK(count,expr)

Function Description

The BENCHMARK() function is used to repeatedly run the expression expr count times. It can be used to time the GBase 8a MPP Cluster to process expressions, and the result is usually 0. When used by the gccli client, it will return the time required for query execution.

Example

Repeat 'hello' 1000000 times.

```
gbase> SELECT BENCHMARK(1000000,'hello') FROM dual;
+----------------------------+
| BENCHMARK(1000000,'hello') |
+----------------------------+
|                          0 |
+----------------------------+
1 row in set
```

explain

The reported time is the time of the client operation, not the CPU time of the server. Calculate the time by executing BENCHMARK() multiple times, and pay attention to interpreting the results by referring to the server's load.

5.1.5.6.3.2 CHARSET(str)

Function Description

Returns the character set used for string parameters.

Example

Example 1: Returns the character set used in the 'Example'.

```
Gbase>SELECT CHARSET ('Example ') from dual;
+-------------------+
|CHARSET ('Example ')|
+-------------------+
| utf8mb4           |
```

```
+--------------------+
1 row in set
```

Example 2: Returns the character set used by 'USER()'.

```
gbase> SELECT CHARSET(USER()) FROM dual;
+------------------+
| CHARSET(USER()) |
+------------------+
| utf8mb4                |
+------------------+
1 row in set
```

5.1.5.6.3.3 COLLATION(str)

Function Description

Returns the character set sorting rule for string parameters.

Example

Example 1: Returns the character set sorting rule for 'abc'.

```
gbase> SELECT COLLATION('abc') FROM dual;
+--------------------+
| COLLATION('abc')      |
+--------------------+
| utf8mb4_ general_ ci |
+--------------------+
1 row in set
```

Example 2: Returns the character set sorting rule for "_gb2312 'abc'".

```
gbase> SELECT COLLATION(_gb2312 'abc') FROM dual;
+--------------------------+
| COLLATION(_gb2312 'abc') |
+--------------------------+
| utf8mb4_ general_ ci             |
+--------------------------+
1 row in set
```

5.1.5.6.3.4 CONNECTION_ ID()

Function Description

Returns the connection ID (thread ID) of the current user. Each connection has a unique ID.

Example

Returns the connection ID of the current user.

```
gbase> SELECT CONNECTION_ ID() FROM dual;
+-----------------+
| CONNECTION_ ID() |
+-----------------+
|                27 |
+-----------------+
1 row in set
```

5.1.5.6.3.5 CURRENT_ USER()

Function Description

Returns the combination of username and host name that matches the current session connection. This value corresponds to the account that determines the user's access permissions.

Example

Example 1: Returns the username and host name that match the current session connection.

```
gbase> SELECT USER() FROM dual;
+---------------------+
| USER()              |
+---------------------+
| root@172.168.83.11  |
+---------------------+
1 row in set
```

Example 2: Returns the combination of username and host name matching the current session connection.

```
gbase> SELECT CURRENT_ USER() FROM dual;
+----------------+
| CURRENT_ USER() |
+----------------+
| root@%         |
+----------------+
1 row in set
```

5.1.5.6.3.6 DATABASE()

Function Description

Returns the database name currently in use.

Example

Example 1: The current database used is' test '.

```
gbase> SELECT DATABASE() FROM dual;
+------------+
| DATABASE() |
+------------+
| test       |
+------------+
1 row in set
```

Example 2: If there is no current database, the system displays a prompt message.

```
gbase> SELECT DATABASE() FROM dual;
ERROR 1046 (3D000): No database selected
```

5.1.5.6.3.7 SESSION_ USER()

Function Description

SESSION_ USER() is equivalent to USER().

Example

Example 1: Returns the current GBase 8a MPP Cluster user and host name.

```
gbase> SELECT SESSION_ USER() FROM dual;
+---------------------+
| SESSION_ USER()     |
+---------------------+
| root@172.168.83.11  |
+---------------------+
1 row in set
```

5.1.5.6.3.8 SYSTEM_ USER()

Function Description

SYSTEM_ USER() is equivalent to USER().

Example

Returns the current GBase 8a MPP Cluster user and host name.

```
gbase> SELECT SYSTEM_ USER() FROM dual;
+---------------------+
| SYSTEM_ USER()      |
+---------------------+
| root@172.168.83.11  |
+---------------------+
1 row in set
```

5.1.5.6.3.9 USER()

Function Description

Returns the current GBase 8a MPP Cluster user and host name.

Example

Example 1: The current user is' root 'and the host name is' 172.168.83.11'.

```
gbase> SELECT USER() FROM dual;
+---------------------+
| USER()              |
+---------------------+
| root@172.168.83.11  |
+---------------------+
1 row in set
```

explain

This value is the username and host name of the user's connection. It is different from Current_ The return value of USER().

Example 2: Users can be reduced to only the username.

```
gbase> SELECT SUBSTRING_ INDEX(USER(),'@',1) FROM dual;
+-------------------------------+
| SUBSTRING_ INDEX(USER(),'@',1) |
+-------------------------------+
| root                          |
+-------------------------------+
1 row in set
```

Example 3: USER() returns values that belong to the UTF8 character set (if an installation package with the GBK character set is selected during installation, values that belong to the GBK character set are returned), so the user also ensures that the '@' string literal can be interpreted in that character set.

```
gbase> SELECT SUBSTRING_ INDEX(USER(),'@',1) FROM dual;
+--------------------------------+
| SUBSTRING_ INDEX(USER(),'@',1) |
+--------------------------------+
| root                           |
+--------------------------------+
1 row in set
```

5.1.5.6.3.10 VERSION()

Function Description

Returns the version number of the cluster as a string.

Example

The version number of the currently used cluster is 9.5.3.17.

```
gbase> SELECT version() FROM dual;
+-----------------+
| version()       |
+-----------------+
| 9.5.3.17.117651 |
+-----------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.5.6.4 auxiliary function

5.1.5.6.4.1 FORMAT(X,D)

Function Description

Format the number X in the form of '#, # #, # #. # #', rounded to the nearest D decimal place. If D is 0, the returned result will have no decimal point or part.

Example

Example 1: Format "12332.123456" to the nearest 4 decimal places.

```
gbase> SELECT FORMAT(12332.123456, 4) FROM dual;
+-------------------------+
```

```
| FORMAT(12332.123456, 4) |
+-------------------------+
| 12,332.1235             |
+-------------------------+
1 row in set
```

Example 2: Format "12332.1" and round it to 4 decimal places. If the decimal part is less than 4 decimal places, fill it with 0.

```
gbase> SELECT FORMAT(12332.1,4) FROM dual;
+------------------+
| FORMAT(12332.1,4) |
+------------------+
| 12,332.1000      |
+------------------+
1 row in set
```

Example 3: If D is 0, the returned result will have no decimal point or part.

```
gbase> SELECT FORMAT(12332.2,0) FROM dual;
+------------------+
| FORMAT(12332.2,0) |
+------------------+
| 12,332           |
+------------------+
1 row in set
```

5.1.5.6.4.2 INET_ ATON(expr)

Function Description

Given a string of network addresses separated by '.', that is, IP addresses, the function returns an integer to represent the address value. The address may be 4 to 8 bytes long.

Example

Example 1: Returns the integer corresponding to "172.168.83.11".

```
gbase> SELECT INET_ ATON('11.83.168.172') FROM dual;
+----------------------------+
| INET_ ATON('11.83.168.172') |
+----------------------------+
|                  190032044 |
+----------------------------+
1 row in set
```

explain

The generated numbers are always in network byte order. As shown in Example 1, the value follows

$11*256^3+83*256^2+168*256^1+172*256^0$ 来计算。

Example 2: INET_ ATON() can also use short format IP addresses.

```
gbase> SELECT INET_ ATON('127.0.0.1'), INET_ ATON('127.1') FROM dual;
+------------------------+--------------------+
| INET_ ATON('127.0.0.1') | INET_ ATON('127.1') |
+------------------------+--------------------+
|                 2130706433 |          2130706433 |
+------------------------+--------------------+
1 row in set
```

5.1.5.6.4.3 INET_ NTOA(expr)

Function Description

Given a numeric network address (4 or 8 bytes), returns the address represented in a dot group style as a string.

Example

Example 1: Returns the IP address corresponding to "190032044".

```
gbase> SELECT INET_ NTOA(190032044) FROM dual;
+----------------------+
| INET_ NTOA(190032044) |
+----------------------+
| 11.83.168.172        |
+----------------------+
1 row in set
```

5.1.5.6.4.4 SLEEP(duration)

Function Description

The sleep (pause) time is the number of seconds given by the duration parameter, and then returns 0.

Example

Example 1: Pause for 10 seconds and return to 0.

```
gbase> SELECT SLEEP(10) FROM dual;
+-----------+
| SLEEP(10) |
+-----------+
|         0 |
+-----------+
1 row in set
```

5.1.5.6.4.5 UUID()

Function Description

Returns a Universal Unique Identifier (UUID) generated based on the "DCE1.1: Remote Procedure Call" CAE (Common Application Environment) manual published by the public organization in October 1997 (document number C706).

UUID is a globally unique number in space and time. Two calls to UUID() will return two different values, even if these calls occur on two independent computers and are not related to each other.

UUID is a 128 bit number represented by a string of five hexadecimal numbers in the format aaaaaa bbbbb cccc dddd eeeeeeeeeee.

The first three digits are generated from the timestamp.

The fourth number protects the uniqueness of the time when the timestamp loses its uniqueness (for example, due to daylight saving time).

The fifth number is an IEEE802 node number that provides spatial uniqueness. If the latter part is not available, use a random number instead (for example, because the host does not have an Ethernet card or does not know how to find the machine address of the interface on the operating system), in which case spatial uniqueness cannot be guaranteed. Nevertheless, the possibility of conflict is still very low.

Currently, only MAC addresses are considered on Linux. On other operating systems, GBase 8a MPP Cluster uses a randomly generated 48 digit number.

Example

Returns a universally unique identifier.

```
gbase> SELECT UUID() FROM dual;
+--------------------------------------+
| UUID()                               |
+--------------------------------------+
| af275192-740f-11ea-a1ca-58a946632074 |
+--------------------------------------+
1 row in set
```

5.1.5.7 Functions and modifiers for the GROUP BY clause

5.1.5.7.1 GROUP BY function

summary

If a user uses an aggregation function in a statement without using the GROUP BY clause, it is equivalent to all data being grouped together.

GBase 8a MPP Cluster extends the usage of GROUP BY:

In a SELECT expression, users can use or calculate columns that do not appear in the GROUP BY section, representing any possible values for this group. Users can use it to avoid sorting and grouping unnecessary classification items, which will result in better performance.

Example

Example 1: Press l. lo_ The shipmode column is grouped.

```
gbase> SELECT c.c_ city, l.lo_ shipmode,l.lo_ supplycost,MAX(l.lo_supplycost) FROM ssbm.lineorder l, ssbm.customer c WHERE l.lo_custkey = c.c_ custkey GROUP BY l.lo_ shipmode;
+-------------+--------------+----------------+-------------------------+
| c_ city     | lo_ shipmode | lo_ supplycost | MAX(l. lo_supplycost) |
+-------------+--------------+----------------+-------------------------+
| JORDAN 9 | TRUCK | 74711 | 125939 |
| JORDAN 9 | REG AIR | 99822 | 125939 |
| EGYPT 6 | AIR | 104928 | 125939 |
| VIETNAM 2 | MAIL | 88770 | 125939 |
| PERU 4 | SHIP | 107205 | 125939 |
| INDONESIA1 | RAIL | 57301 | 125939 |
| EGYPT 6 | FOB | 96210 | 125939 |
+-------------+--------------+----------------+-------------------------+
7 rows in set
```

be careful

If the column omitted by the user in the GROUP BY section is not unique in the group, please do not use this feature, otherwise unpredictable results will be obtained.

5.1.5.7.1.1 AVG([DISTINCT] expr)

Function Description

Returns the average value of expr. Using the DISTINCT option, you can return the average of all different values in expr.

Example

Example 1: Returns the average value of 'lo_supplycost'.

```
gbase> SELECT lo_ orderpriority, AVG(lo_supplycost) FROM
ssbm.lineorder GROUP BY LO_ ORDERPRIORITY;
+-----------------+--------------------+
| lo_ orderpriority | AVG(lo_supplycost) |
+-----------------+--------------------+
| 2-HIGH                  |              89935.4667 |
| 1-URGENT              |               89979.8854 |
| 4-NOT SPECI         |              89965.0583 |
| 5-LOW                  |               89978.3022 |
| 3-MEDIUM             |               89954.0913 |
+-----------------+--------------------+
5 rows in set
```

5.1.5.7.1.2 COUNT(expr)

Function Description

Returns the total number of records with non NULL values in a row retrieved by a SELECT statement.

Example

Example 1: Retrieve the total number of records that meet the criteria and press c_ Grouping with mktsegments.

```
gbase> SELECT c.c_ mktsegment,COUNT(*) FROM ssbm.lineorder l,
ssbm.customer c WHERE l.lo_ custkey = c.c_ custkey GROUP BY c.c_
mktsegment;
+--------------+----------+
| c_ mktsegment | COUNT(*) |
+--------------+----------+
| HOUSEHOLD      |   1217475 |
| AUTOMOBILE     |   1174124 |
| BUILDING       |   1230857 |
| FURNITURE      |   1182372 |
| MACHINERY      |   1196343 |
+--------------+----------+
5 rows in set
```

 **explain**

COUNT (*) is slightly different in the number of retrieved record rows it returns, regardless of whether the record rows include NULL values. Because SSBM is a randomly generated demonstration database, the execution results may vary, and the above results are only a demonstration result.

5.1.5.7.1.3 COUNT(DISTINCT expr,[expr...])

Function Description

Returns the total number of different non NULL values.

Example

Example 1: Return lo_ The total number of different non NULL values in the orderkey column.

```
gbase> SELECT COUNT(DISTINCT lo_orderkey) FROM ssbm.lineorder;
+----------------------------+
| COUNT(DISTINCT lo_orderkey) |
+----------------------------+
|                    1500000 |
+----------------------------+
1 row in set
```

explain

In GBase 8a MPP Cluster, users obtain the number of different expression combinations that do not contain NULL by giving a list of expressions.

5.1.5.7.1.4 MIN(), MAX()

Function Description

MIN ([DISTINCT] expr), MAX ([DISTINCT] expr) returns the minimum or maximum value of expr. You can set parameters for MIN() and MAX(), in which case they will return the minimum or maximum value of the column specified by the parameter. The DISTINCT keyword can be used to find the minimum or maximum values of different values for expr, however, this results in the same results as omitting DISTINCT. MIN and MAX do not include NULL values.

Example

Example 1: Return lo_ The maximum and minimum values of the supplycost column.

```
gbase> SELECT lo_ shipmode,MAX(lo_supplycost),MIN(lo_supplycost)
FROM ssbm.lineorder GROUP BY lo_ shipmode;
+-------------+--------------------+--------------------+
| lo_ shipmode | MAX(lo_supplycost) | MIN(lo_supplycost) |
+-------------+--------------------+--------------------+
| AIR          |             125939 |              54060 |
| REG AIR      |             125939 |              54060 |
| TRUCK        |             125939 |              54060 |
| SHIP         |             125939 |              54060 |
| MAIL         |             125939 |              54060 |
| FOB          |             125939 |              54060 |
| RAIL         |             125939 |              54060 |
+-------------+--------------------+--------------------+
7 rows in set
```

5.1.5.7.1.5 SUM([DISTINCT] expr)

Function Description

Returns the sum of expr. If there are no record rows in the returned result set, it will return NULL. The DISTINCT keyword is used to calculate the sum of different values in expr.

5.1.5.7.1.6 GROUP_ CONCAT(expr)

grammar

This function supports concatenation of clustered column strings within the same group, and its complete syntax is as follows:

```
GROUP_ CONCAT ( [distinct]
expr
[order by …[asc/desc]]
[topN xxxx]
[separator 'xxxx']
)
```

Table 530 Parameter Description

Parameter Name	Description
distinct	Remove duplicate values from clustered columns within the same group, optional. Default value: Keep duplicate values.
expr	Aggregated columns: Column based expressions that support multiple types. Required.
[order by …[asc/desc]]	The values of the clustered columns within the same group will be output in the order of sorting. Optional, supports multi column sorting lists, and can specify the ascending and descending order of the sorting sequence. Default value: If no sorting list is entered, it will be sorted in ascending order according to the string sorting rule of the clustered column.
[topN xxxx]	Represents the maximum number of row values that can output aggregated columns within the same group, optional. Default value: If no topN value is entered, all row values of the clustered column within the same group will be output. Limit value: topN represents the maximum number of connection strings in a row, which determines the memory size and other related information allocated by the subsequent engine calculations. It should not be set too large. So, during parsing, there are restrictions on it, such that when an element has only one character limit, the topN does not exceed group_ concat_ max_ Len (in bytes, default value is 1024) is the length specified by the environment variable. The default character set for 952 is utf8mb4 (4 bytes), while the default character set for 862 is utf8 (3 bytes) group_ concat_ max_ Len is the longest number of bytes. When topN is 0, that is, group_ Concat (colname, topN 0) indicates that no rows are clustered, group_ The value of the concat function is null.

Parameter Name	Description
[separator 'xxxx']	The separator between row values of clustered columns within the same group. Optional. Default value: If no separator is entered, the default separator is a half width comma. Limit value: unlimited length, but GROUP_ CONCAT output result exceeds group_ concat_ max_ Len will report an error.

be careful

- The distinct parameter and order by are mutually exclusive.
- When topN is 0, that is, group_ Concat (colname, topN 0), do not aggregate any rows, group_ The value of the concat function is null.
- When there is no topN, i.e. group_ When concat (colname), there is no restriction on clustered rows.
- The default character set for 952 is utf8mb4 (4 bytes), while the default character set for 862 is utf8 (3 bytes). group_ concat_ max_ Len is the longest number of bytes.
- group_ Blob type parameters are not supported in concat.

5.1.5.7.1.7 COVAR_ POP()

Function Description

Returns the overall covariance of a pair of expressions. The returned result is of the double data type.

grammar

```
COVAR_ POP( expression1, expression2)
```

Table 531 Parameter Description

Parameter Name	Description
expression1	Numeric expression
expression2	Numeric expression

be careful

COVAR_ The POP function calculation will ignore records with null values for expression1 or expression2.

Example

```
create table all_ tables(owner int, avg_row_len int, avg_space int);

insert into all_ tables values(1, 1241,          2446);
insert into all_ tables values(1, 1158,          1028);
insert into all_ tables values(1,  332,           621);
insert into all_ tables values(1,  126,           408);
insert into all_ tables values(1,  173,          1222);
insert into all_ tables values(1,  180,           834);
insert into all_ tables values(1,   96,           702);
insert into all_ tables values(1,  285,           158);
insert into all_ tables values(1,  null,          159);
insert into all_ tables values(1,  190, null);
insert into all_ tables values(1, null,null);
insert into all_ tables values(2, 1,           2);
insert into all_ tables values(2, 3,           4);
insert into all_ tables values(2,  5,          6);
gbase> SELECT owner,covar_ pop(avg_row_len, avg_space) from all_ tables
group by owner;
+-------+-----------------------------------+
| owner | covar_ pop(avg_row_len, avg_space) |
+-------+-----------------------------------+
|     2 |                        2.66666667 |
|     1 |                    203404.29687500 |
+-------+-----------------------------------+
2 rows in set
```

5.1.5.7.1.8 COVAR_ SAMP()

Function Description

Returns the sample covariance of a pair of expressions. The returned result is of the double data type.

grammar

```
COVAR_ SAMP( expression1, expression2)
```

Parameter Description

Two parameter expressions must be specified, expression1 and expression2 must be numerical expressions.

be careful

COVAR_ The SAMP function calculation will ignore records with null values for expression1 or expression2.

Example

```
gbase> SELECT owner,covar_ samp(avg_row_len, avg_space) as covar from
all_ tables group by owner;
+-------+----------------------+
| owner | covar                |
+-------+----------------------+
|     1 |  232462.053571428571 |
|     2 |       4.000000000000 |
+-------+----------------------+
2 rows in set
```

5.1.5.7.1.9 CORR()

Function Description

Returns the correlation coefficient of a pair of expressions. The returned result is of the double data type.

grammar

```
CORR( expression1, expression2)
```

Parameter Description

Two parameter expressions must be specified, expression1 and expression2 must be numerical expressions.

be careful

The CORR function calculation will ignore records with a NULL value of expression1 or expression2, and requires at least two rows of record aggregation calculations. Otherwise, a NULL value will be returned.

Example

Example 1:

```
create table sales (quantity int, commission int);
insert into sales values(1,2),(3,4),(5,6);
gbase> SELECT corr(quantity, commission) from sales;
+----------------------------+
| corr(quantity, commission) |
+----------------------------+
|                0.99999999975 |
+----------------------------+
1 row in set
```

Example 2:

```
create table data(max_entents int, max_trans int, initial_extent int);

insert into data values(1, 1241,          2446);
insert into data values(1, 1158,          1028);
insert into data values(1,  332,           621);
insert into data values(1,  126,           408);
insert into data values(1,  173,          1222);
insert into data values(1,  180,           834);
insert into data values(1,   96,           702);
insert into data values(1,  285,           158);
insert into data values(1,  null,          159);
insert into data values(1,  190, null);
insert into data values(1, null,null);
insert into data values(2, 1,          2);
insert into data values(2, 3,          4);
insert into data values(2,  5,          6);


gbase> SELECT max_ entents, corr(max_trans, initial_extent) from data
group by max_ entents;
+-------------+--------------------------------+
| max_ entents | corr(max_trans, initial_extent) |
+-------------+--------------------------------+
|           2 |                 0.99999999975 |
|           1 |              0.707398407157965 |
+-------------+--------------------------------+
2 rows in set
```

5.1.5.7.1.10 BIT_ AND(expr)

Function Description

Perform a bitwise AND operation on all bits in the returned expr. Calculations are performed with 64 bit (BIGINT) precision. If there are no matching rows, the function

returns 18446744073709551615 (the value is an unsigned BIGINT type with all bits set to 1).

5.1.5.7.1.11 BIT_ OR(expr)

Function Description

Perform bitwise OR operations on all bits in the returned expr. Calculations are performed with 64 bit (BIGINT) precision. If there are no matching rows, the function returns 0.

5.1.5.7.1.12 BIT_ XOR(expr)

Function Description

Returns the XOR values of all bits in expr bit by bit. Calculate using 64 bit precision (BIGINT). If there are no matching rows, the function returns 0.

5.1.5.7.1.13 STD(expr), STDDEV(expr)

Function Description

Returns the standard deviation of expr.

be careful

Due to differences between databases, the STDDEV() function of GBase 8a MPP Cluster does not behave consistently with the Oracle STDDEV() function.

5.1.5.7.1.14 STDDEV_ POP(expr)

Function Description

Returns the overall standard deviation of expr (i.e. the square root of VAR_POP()), which users can use STD() or STDDEV() instead, but not standard SQL.

5.1.5.7.1.15 STDDEV_ SAMP(expr)

Function Description

Returns the sample standard deviation of expr (which is the square root of VAR_SAMP()), which is equivalent to Oracle's stddev (expr) function.

5.1.5.7.1.16 VAR_ POP(expr)

Function Description

var_ The pop function calculates the overall variance of the data within the window, and the formula for calculating the overall variance is:

$$s^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

In the formula,

The sample is a random vector: (x1, x2,... xn)

The mean of the sample is: $\overline{x}$

the variance of the sample is: s2

var_ The pop function requires a numeric parameter, which can be a constant, expression, field, or NULL value. The supported types include int, decimal, and real, while other types may report errors. The NULL value in the sample is ignored during calculation, and the population variance with less than 1 sample size is NULL.

Example

gbase> SELECT *, var_ pop(totalamount) over (partition by uname order by dt) as var_ pop from tt;

```
+----+------------+-------+-------------+------------------+
| id | dt         | uname | totalamount | var_ pop         |
+----+------------+-------+-------------+------------------+
|  2 | 2016-06-05 | A     |         148 |            42.25 |
|  1 | 2016-06-05 | A     |         135 |            42.25 |
|  4 | 2016-06-02 | B     |         153 |           272.25 |
|  3 | 2016-06-02 | B     |         120 |           272.25 |
|  5 | 2016-06-10 | B     |         198 |             1022 |
|  8 | 2016-02-05 | C     |        NULL |             NULL |
|  6 | 2016-08-05 | C     |         201 |                0 |
|  9 | 2016-08-06 | C     |        NULL |                0 |
|  7 | 2016-08-09 | C     |         129 |             1296 |
| 14 | 2016-07-02 | D     |         172 |                0 |
| 13 | 2016-09-01 | D     |         165 |            12.25 |
| 15 | NULL       | D     |         149 | 92.6666666666667 |
| 10 | 2016-06-01 | NULL  |         125 |                0 |
| 11 | 2016-07-02 | NULL  |         131 |                9 |
| 12 | 2016-08-03 | NULL  |         152 |              134 |
+----+------------+-------+-------------+------------------+
```

5.1.5.7.1.17 VAR_ SAMP(expr)

Function Description

var_ The sample variance of the data in the samp function calculation window, and the calculation formula of the sample variance is:

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

In the formula,

The sample is a random vector: (x1, x2,... xn)

The mean of the sample is: $\overline{x}$

the variance of the sample is: s2

var_ The samp function requires a numeric parameter, which can be a constant, expression, field, or NULL value. The supported types include int, decimal, and real, while other types may report errors. The NULL value in the sample is ignored during calculation, and the sample variance with the number of samples less than or equal to 1 is NULL.

Example

gbase> SELECT *, var_ samp(totalamount) over (partition by uname order by dt) as var_ samp from tt;

```
+----+------------+-------+-------------+----------+
| id | dt         | uname | totalamount | var_ samp |
+----+------------+-------+-------------+----------+
|  2 | 2016-06-05 | A     |         148 |     84.5 |
|  1 | 2016-06-05 | A     |         135 |     84.5 |
|  4 | 2016-06-02 | B     |         153 |    544.5 |
|  3 | 2016-06-02 | B     |         120 |    544.5 |
|  5 | 2016-06-10 | B     |         198 |     1533 |
|  8 | 2016-02-05 | C     |        NULL |     NULL |
|  6 | 2016-08-05 | C     |         201 |     NULL |
|  9 | 2016-08-06 | C     |        NULL |     NULL |
|  7 | 2016-08-09 | C     |         129 |     2592 |
| 14 | 2016-07-02 | D     |         172 |     NULL |
| 13 | 2016-09-01 | D     |         165 |     24.5 |
| 15 | NULL       | D     |         149 |      139 |
| 10 | 2016-06-01 | NULL  |         125 |     NULL |
| 11 | 2016-07-02 | NULL  |         131 |       18 |
| 12 | 2016-08-03 | NULL  |         152 |      201 |
+----+------------+-------+-------------+----------+
```

5.1.5.7.1.18 VARIANCE(expr)

Function Description

Returns the standard variance of expr. This is an extension to standard SQL. Standard SQL Function VAR_ POP() can be used instead.

Example

Example 1:

```
gbase> SELECT `users`.u_ name , VARIANCE(orders.amount) FROM users,
orders WHERE users.u_ id = orders.u_ id GROUP BY `users`.u_ name
ORDER BY    `users`.u_ name LIMIT 2;
+------------+-------------------------+
| u_ name          | VARIANCE(orders.amount) |
+------------+-------------------------+
| li                 |              1979296.00000000 |
| qian             |                735555.55555556 |
+------------+-------------------------+
2 rows in set (Elapsed: 00:00:00.47)
```

be careful

If the column omitted by the user in the GROUP BY section is not unique in the group, please do not use this feature, otherwise unpredictable results will be obtained.

GROUP BY optimization

Optimize GROUP BY using hint:

```
create table users(u_id int,u_name char(10),birthday datetime,address
varchar(10));
create table products(p_id int,p_name char(10),price float,stocks int);
create table orders(o_id int,u_id int,p_id int,amount int);
Insert into users values (0, 'zhao', '1990-12-29 12:00:00', 'Jilin, Changchun,
China');
Insert into users values (1, 'qian', '1990-12-29 12:00:00', 'Beijing, China');
Insert into users values (2, 'sun', '1990-12-29 12:00:00', 'Shanghai, China');
Insert into users values (3, 'li', '1990 12 29 12:00:00', 'Tianjin, China');
Insert into users values (4, 'zhou', '1990-12-29 12:00:00', 'Wuhan, Hubei,
China');
Insert into users values (5, 'wu', '1990-12-29 12:00:00', 'Changsha, Hunan,
China');
Insert into users values (6, 'zheng', '1990-12-29 12:00:00', 'Cangzhou, Hebei,
China');
Insert into users values (7, 'wang', '1990-12-29 12:00:00', 'Liaoning,
Shenyang, China');
Insert into users values (8, 'zhao', '1990-12-29 12:00:00', 'Yunnan, Kunming,
China');
Insert into users values (9, 'li', '1990-12-29 12:00:00', 'China, Tibet, Lhasa');
insert into orders values (600001,1,100010,2500);
insert into orders values (600002,3,100008,3500);
insert into orders values (600003,4,100009,500);
insert into orders values (600004,5,100007,5000);
insert into orders values (600005,9,100003,2510);
insert into orders values (600006,0,100004,10500);
insert into orders values (600007,3,100002,5500);
insert into orders values (600008,3,100001,6000);
insert into orders values (600009,4,100009,2100);
insert into orders values (600010,5,100001,1300);
insert into orders values (600011,6,100007,8900);
insert into orders values (600012,9,100008,2900);
insert into orders values (600013,2,100009,6900);
insert into orders values (600014,1,100002,3600);
insert into orders values (600015,1,100003,1500);
SELECT /*+ first_ groupby */ distinct `users`.u_ name , sum(orders.amount)
FROM users, orders WHERE users.u_ id = orders.u_ id GROUP BY
`users`.u_ name ORDER BY `users`.u_ name limit 3;
```

The above SQL will first perform group by on the orders table before joining users.

5.1.5.7.1.19 CUME_ DIST

Function Description

cume_ The dist function is used to count the number of rows that are less than or equal to the current value/the total number of rows in the window. For example, it can be used to count the proportion of the number of people who are less than or equal to the current salary to the total number of people.

cume_ The dist function does not require parameters.

Example

```
gbase> SELECT *, cume_ dist() over (partition by uname order by dt) as
cume_ dist from tt;
+----+------------+-------+-------------+-------------------+
| id | dt         | uname | totalamount | cume_ dist        |
+----+------------+-------+-------------+-------------------+
|  2 | 2016-06-05 | A     |         148 |                 1 |
|  1 | 2016-06-05 | A     |         135 |                 1 |
|  4 | 2016-06-02 | B     |         153 | 0.666666666666667 |
|  3 | 2016-06-02 | B     |         120 | 0.666666666666667 |
|  5 | 2016-06-10 | B     |         198 |                 1 |
|  8 | 2016-02-05 | C     |        NULL |              0.25 |
|  6 | 2016-08-05 | C     |         201 |               0.5 |
|  9 | 2016-08-06 | C     |        NULL |              0.75 |
|  7 | 2016-08-09 | C     |         129 |                 1 |
| 14 | 2016-07-02 | D     |         172 | 0.333333333333333 |
| 13 | 2016-09-01 | D     |         165 | 0.666666666666667 |
| 15 | NULL       | D     |         149 |                 1 |
| 10 | 2016-06-01 | NULL  |         125 | 0.333333333333333 |
| 11 | 2016-07-02 | NULL  |         131 | 0.666666666666667 |
| 12 | 2016-08-03 | NULL  |         152 |                 1 |
+----+------------+-------+-------------+-------------------+
```

5.1.5.7.1.20 NTILE

Function Description

The ntile function is used to sequentially divide grouped data into n slices and return the current slice value. If the slices are uneven, the distribution of the previous slices will be increased by default.

The ntile function supports a constant parameter, supports NULL values, and currently does not support fields. The parameter range is an integer greater than 0. If the input is a constant string, convert it to an integer as parameter input (note: the string is converted to an integer until the first character from the left that is not a number); If the input is a floating-point type constant, it is rounded as the parameter input.

Example

```
gbase> SELECT *, ntile(2) over (partition by uname order by dt) as ntile
from tt;
+----+------------+-------+-------------+-------+
| id | dt         | uname | totalamount | ntile |
+----+------------+-------+-------------+-------+
|  2 | 2016-06-05 | A     |         148 |     1 |
|  1 | 2016-06-05 | A     |         135 |     2 |
|  4 | 2016-06-02 | B     |         153 |     1 |
|  3 | 2016-06-02 | B     |         120 |     1 |
|  5 | 2016-06-10 | B     |         198 |     2 |
|  8 | 2016-02-05 | C     |        NULL |     1 |
|  6 | 2016-08-05 | C     |         201 |     1 |
|  9 | 2016-08-06 | C     |        NULL |     2 |
|  7 | 2016-08-09 | C     |         129 |     2 |
| 14 | 2016-07-02 | D     |         172 |     1 |
| 13 | 2016-09-01 | D     |         165 |     1 |
| 15 | NULL       | D     |         149 |     2 |
| 10 | 2016-06-01 | NULL  |         125 |     1 |
| 11 | 2016-07-02 | NULL  |         131 |     1 |
| 12 | 2016-08-03 | NULL  |         152 |     2 |
+----+------------+-------+-------------+-------+
```

```
gbase> SELECT *, ntile('2') over (partition by uname order by dt) as ntile
from tt;
+----+------------+-------+-------------+-------+
| id | dt         | uname | totalamount | ntile |
+----+------------+-------+-------------+-------+
|  2 | 2016-06-05 | A     |         148 |     1 |
|  1 | 2016-06-05 | A     |         135 |     2 |
|  4 | 2016-06-02 | B     |         153 |     1 |
|  3 | 2016-06-02 | B     |         120 |     1 |
|  5 | 2016-06-10 | B     |         198 |     2 |
|  8 | 2016-02-05 | C     |        NULL |     1 |
|  6 | 2016-08-05 | C     |         201 |     1 |
|  9 | 2016-08-06 | C     |        NULL |     2 |
|  7 | 2016-08-09 | C     |         129 |     2 |
| 14 | 2016-07-02 | D     |         172 |     1 |
| 13 | 2016-09-01 | D     |         165 |     1 |
| 15 | NULL       | D     |         149 |     2 |
| 10 | 2016-06-01 | NULL  |         125 |     1 |
| 11 | 2016-07-02 | NULL  |         131 |     1 |
| 12 | 2016-08-03 | NULL  |         152 |     2 |
+----+------------+-------+-------------+-------+
gbase> SELECT *, ntile(2.1) over (partition by uname order by dt) as ntile
from tt;
+----+------------+-------+-------------+-------+
| id | dt         | uname | totalamount | ntile |
+----+------------+-------+-------------+-------+
|  2 | 2016-06-05 | A     |         148 |     1 |
|  1 | 2016-06-05 | A     |         135 |     2 |
|  4 | 2016-06-02 | B     |         153 |     1 |
|  3 | 2016-06-02 | B     |         120 |     1 |
|  5 | 2016-06-10 | B     |         198 |     2 |
|  8 | 2016-02-05 | C     |        NULL |     1 |
|  6 | 2016-08-05 | C     |         201 |     1 |
|  9 | 2016-08-06 | C     |        NULL |     2 |
|  7 | 2016-08-09 | C     |         129 |     2 |
| 14 | 2016-07-02 | D     |         172 |     1 |
| 13 | 2016-09-01 | D     |         165 |     1 |
| 15 | NULL       | D     |         149 |     2 |
| 10 | 2016-06-01 | NULL  |         125 |     1 |
| 11 | 2016-07-02 | NULL  |         131 |     1 |
| 12 | 2016-08-03 | NULL  |         152 |     2 |
+----+------------+-------+-------------+-------+
```

5.1.5.7.1.21 FIRST_ VALUE

Function Description

first_ The value function takes the first value of the current row after sorting within the window.

first_ The value function supports one parameter and can input constants, fields, etc.

Example

```
gbase> SELECT *, first_ value(totalamount) over (partition by uname order by dt) as first_ value from tt;
+----+------------+-------+-------------+-------------+
| id | dt         | uname | totalamount | first_ value |
+----+------------+-------+-------------+-------------+
|  2 | 2016-06-05 | A     |         148 |         148 |
|  1 | 2016-06-05 | A     |         135 |         148 |
|  4 | 2016-06-02 | B     |         153 |         153 |
|  3 | 2016-06-02 | B     |         120 |         153 |
|  5 | 2016-06-10 | B     |         198 |         153 |
|  8 | 2016-02-05 | C     |        NULL |        NULL |
|  6 | 2016-08-05 | C     |         201 |        NULL |
|  9 | 2016-08-06 | C     |        NULL |        NULL |
|  7 | 2016-08-09 | C     |         129 |        NULL |
| 14 | 2016-07-02 | D     |         172 |         172 |
| 13 | 2016-09-01 | D     |         165 |         172 |
| 15 | NULL       | D     |         149 |         172 |
| 10 | 2016-06-01 | NULL  |         125 |         125 |
| 11 | 2016-07-02 | NULL  |         131 |         125 |
| 12 | 2016-08-03 | NULL  |         152 |         125 |
+----+------------+-------+-------------+-------------+
```

```
gbase> SELECT *, first_ value('const') over (partition by uname order by dt)
as first_ value from tt;
+----+------------+-------+-------------+-------------+
| id | dt         | uname | totalamount | first_ value |
+----+------------+-------+-------------+-------------+
|  2 | 2016-06-05 | A     |         148 | const       |
|  1 | 2016-06-05 | A     |         135 | const       |
|  4 | 2016-06-02 | B     |         153 | const       |
|  3 | 2016-06-02 | B     |         120 | const       |
|  5 | 2016-06-10 | B     |         198 | const       |
|  8 | 2016-02-05 | C     |        NULL | const       |
|  6 | 2016-08-05 | C     |         201 | const       |
|  9 | 2016-08-06 | C     |        NULL | const       |
|  7 | 2016-08-09 | C     |         129 | const       |
| 14 | 2016-07-02 | D     |         172 | const       |
| 13 | 2016-09-01 | D     |         165 | const       |
| 15 | NULL       | D     |         149 | const       |
| 10 | 2016-06-01 | NULL  |         125 | const       |
| 11 | 2016-07-02 | NULL  |         131 | const       |
| 12 | 2016-08-03 | NULL  |         152 | const       |
+----+------------+-------+-------------+-------------+
gbase> SELECT *, first_ value(NULL) over (partition by uname order by dt)
as first_ value from tt;
+----+------------+-------+-------------+-------------+
| id | dt         | uname | totalamount | first_ value |
+----+------------+-------+-------------+-------------+
|  2 | 2016-06-05 | A     |         148 |        NULL |
|  1 | 2016-06-05 | A     |         135 |        NULL |
|  4 | 2016-06-02 | B     |         153 |        NULL |
|  3 | 2016-06-02 | B     |         120 |        NULL |
|  5 | 2016-06-10 | B     |         198 |        NULL |
|  8 | 2016-02-05 | C     |        NULL |        NULL |
|  6 | 2016-08-05 | C     |         201 |        NULL |
|  9 | 2016-08-06 | C     |        NULL |        NULL |
|  7 | 2016-08-09 | C     |         129 |        NULL |
| 14 | 2016-07-02 | D     |         172 |        NULL |
| 13 | 2016-09-01 | D     |         165 |        NULL |
| 15 | NULL       | D     |         149 |        NULL |
| 10 | 2016-06-01 | NULL  |         125 |        NULL |
| 11 | 2016-07-02 | NULL  |         131 |        NULL |
| 12 | 2016-08-03 | NULL  |         152 |        NULL |
+----+------------+-------+-------------+-------------+
```

5.1.5.7.1.22 LAST_ VALUE

Function Description

last_ Value Function and First_ The value function, on the contrary, takes the last value of the current row after sorting within the window.

last_ The value function also supports one parameter and can input constants, fields, etc.

Example

```
gbase> SELECT *, last_ value(totalamount) over (partition by uname order by dt) as last_ value from tt;
+----+------------+-------+-------------+------------+
| id | dt         | uname | totalamount | last_ value |
+----+------------+-------+-------------+------------+
|  2 | 2016-06-05 | A     |         148 |        135 |
|  1 | 2016-06-05 | A     |         135 |        135 |
|  4 | 2016-06-02 | B     |         153 |        120 |
|  3 | 2016-06-02 | B     |         120 |        120 |
|  5 | 2016-06-10 | B     |         198 |        198 |
|  8 | 2016-02-05 | C     |        NULL |       NULL |
|  6 | 2016-08-05 | C     |         201 |        201 |
|  9 | 2016-08-06 | C     |        NULL |       NULL |
|  7 | 2016-08-09 | C     |         129 |        129 |
| 14 | 2016-07-02 | D     |         172 |        172 |
| 13 | 2016-09-01 | D     |         165 |        165 |
| 15 | NULL       | D     |         149 |        149 |
| 10 | 2016-06-01 | NULL  |         125 |        125 |
| 11 | 2016-07-02 | NULL  |         131 |        131 |
| 12 | 2016-08-03 | NULL  |         152 |        152 |
+----+------------+-------+-------------+------------+
```

```
gbase> SELECT *, last_ value('const') over (partition by uname order by dt)
as last_ value from tt;
+----+------------+-------+-------------+------------+
| id | dt         | uname | totalamount | last_ value |
+----+------------+-------+-------------+------------+
|  2 | 2016-06-05 | A     |         148 | const      |
|  1 | 2016-06-05 | A     |         135 | const      |
|  4 | 2016-06-02 | B     |         153 | const      |
|  3 | 2016-06-02 | B     |         120 | const      |
|  5 | 2016-06-10 | B     |         198 | const      |
|  8 | 2016-02-05 | C     |        NULL | const      |
|  6 | 2016-08-05 | C     |         201 | const      |
|  9 | 2016-08-06 | C     |        NULL | const      |
|  7 | 2016-08-09 | C     |         129 | const      |
| 14 | 2016-07-02 | D     |         172 | const      |
| 13 | 2016-09-01 | D     |         165 | const      |
| 15 | NULL       | D     |         149 | const      |
| 10 | 2016-06-01 | NULL  |         125 | const      |
| 11 | 2016-07-02 | NULL  |         131 | const      |
| 12 | 2016-08-03 | NULL  |         152 | const      |
+----+------------+-------+-------------+------------+
gbase> SELECT *, last_ value(NULL) over (partition by uname order by dt)
as last_ value from tt;
+----+------------+-------+-------------+------------+
| id | dt         | uname | totalamount | last_ value |
+----+------------+-------+-------------+------------+
|  2 | 2016-06-05 | A     |         148 |       NULL |
|  1 | 2016-06-05 | A     |         135 |       NULL |
|  4 | 2016-06-02 | B     |         153 |       NULL |
|  3 | 2016-06-02 | B     |         120 |       NULL |
|  5 | 2016-06-10 | B     |         198 |       NULL |
|  8 | 2016-02-05 | C     |        NULL |       NULL |
|  6 | 2016-08-05 | C     |         201 |       NULL |
|  9 | 2016-08-06 | C     |        NULL |       NULL |
|  7 | 2016-08-09 | C     |         129 |       NULL |
| 14 | 2016-07-02 | D     |         172 |       NULL |
| 13 | 2016-09-01 | D     |         165 |       NULL |
| 15 | NULL       | D     |         149 |       NULL |
| 10 | 2016-06-01 | NULL  |         125 |       NULL |
| 11 | 2016-07-02 | NULL  |         131 |       NULL |
| 12 | 2016-08-03 | NULL  |         152 |       NULL |
+----+------------+-------+-------------+------------+
```

5.1.5.7.1.23 NTH_ VALUE

Function Description

nth_ The value function is different from first_ Value and Last_ value, nth_ The value function can specify the offset of the value.

nth_ The value function supports two parameters, the first parameter being the same as first_ Value and Last_ The value is exactly the same; The second parameter is the offset, which requires a constant parameter greater than 0 and does not support NULL values and fields.

Example

```
gbase> SELECT *, nth_ value(totalamount, 2) over (partition by uname
order by dt) as nth_ value from tt;
+----+------------+-------+-------------+-----------+
| id | dt         | uname | totalamount | nth_ value |
+----+------------+-------+-------------+-----------+
|  2 | 2016-06-05 | A     |         148 |       135 |
|  1 | 2016-06-05 | A     |         135 |       135 |
|  4 | 2016-06-02 | B     |         153 |       120 |
|  3 | 2016-06-02 | B     |         120 |       120 |
|  5 | 2016-06-10 | B     |         198 |       120 |
|  8 | 2016-02-05 | C     |        NULL |      NULL |
|  6 | 2016-08-05 | C     |         201 |       201 |
|  9 | 2016-08-06 | C     |        NULL |       201 |
|  7 | 2016-08-09 | C     |         129 |       201 |
| 14 | 2016-07-02 | D     |         172 |      NULL |
| 13 | 2016-09-01 | D     |         165 |       165 |
| 15 | NULL       | D     |         149 |       165 |
| 10 | 2016-06-01 | NULL  |         125 |      NULL |
| 11 | 2016-07-02 | NULL  |         131 |       131 |
| 12 | 2016-08-03 | NULL  |         152 |       131 |
+----+------------+-------+-------------+-----------+
gbase> SELECT *, nth_ value(totalamount, NULL) over (partition by uname
order by dt) as nth_ value from tt;
ERROR 1733 (HY000): (GBA-01EX-700) Gbase general error: argument[2] of
nth_ value is out of range
gbase> SELECT *, nth_ value(totalamount, 0) over (partition by uname
order by dt) as nth_ value from tt;
ERROR 1733 (HY000): (GBA-01EX-700) Gbase general error: argument[2] of
nth_ value is out of range

gbase> SELECT *, nth_ value('const', 2) over (partition by uname order by
dt) as nth_ value from tt;
```

gbase> SELECT *, nth_ value(totalamount, 2) over (partition by uname order by dt) as nth_ value from tt;

```
+----+------------+-------+-------------+-----------+
| id | dt         | uname | totalamount | nth_ value |
+----+------------+-------+-------------+-----------+
|  2 | 2016-06-05 | A     |         148 |       135 |
|  1 | 2016-06-05 | A     |         135 |       135 |
|  4 | 2016-06-02 | B     |         153 |       120 |
|  3 | 2016-06-02 | B     |         120 |       120 |
|  5 | 2016-06-10 | B     |         198 |       120 |
|  8 | 2016-02-05 | C     |        NULL |      NULL |
|  6 | 2016-08-05 | C     |         201 |       201 |
|  9 | 2016-08-06 | C     |        NULL |       201 |
|  7 | 2016-08-09 | C     |         129 |       201 |
| 14 | 2016-07-02 | D     |         172 |      NULL |
| 13 | 2016-09-01 | D     |         165 |       165 |
| 15 | NULL       | D     |         149 |       165 |
| 10 | 2016-06-01 | NULL  |         125 |      NULL |
| 11 | 2016-07-02 | NULL  |         131 |       131 |
| 12 | 2016-08-03 | NULL  |         152 |       131 |
+----+------------+-------+-------------+-----------+
```

gbase> SELECT *, nth_ value(totalamount, NULL) over (partition by uname order by dt) as nth_ value from tt;

ERROR 1733 (HY000): (GBA-01EX-700) Gbase general error: argument[2] of nth_ value is out of range

gbase> SELECT *, nth_ value(totalamount, 0) over (partition by uname order by dt) as nth_ value from tt;

ERROR 1733 (HY000): (GBA-01EX-700) Gbase general error: argument[2] of nth_ value is out of range

```
+----+------------+-------+-------------+-----------+
| id | dt         | uname | totalamount | nth_ value |
+----+------------+-------+-------------+-----------+
|  2 | 2016-06-05 | A     |         148 | const      |
|  1 | 2016-06-05 | A     |         135 | const      |
|  4 | 2016-06-02 | B     |         153 | const      |
|  3 | 2016-06-02 | B     |         120 | const      |
|  5 | 2016-06-10 | B     |         198 | const      |
|  8 | 2016-02-05 | C     |        NULL | NULL       |
|  6 | 2016-08-05 | C     |         201 | const      |
|  9 | 2016-08-06 | C     |        NULL | const      |
|  7 | 2016-08-09 | C     |         129 | const      |
| 14 | 2016-07-02 | D     |         172 | NULL       |
| 13 | 2016-09-01 | D     |         165 | const      |
| 15 | NULL       | D     |         149 | const      |
```

```
gbase> SELECT *, nth_ value(totalamount, 2) over (partition by uname
order by dt) as nth_ value from tt;
+----+------------+-------+-------------+-----------+
| id | dt         | uname | totalamount | nth_ value |
+----+------------+-------+-------------+-----------+
|  2 | 2016-06-05 | A     |         148 |       135 |
|  1 | 2016-06-05 | A     |         135 |       135 |
|  4 | 2016-06-02 | B     |         153 |       120 |
|  3 | 2016-06-02 | B     |         120 |       120 |
|  5 | 2016-06-10 | B     |         198 |       120 |
|  8 | 2016-02-05 | C     |        NULL |      NULL |
|  6 | 2016-08-05 | C     |         201 |       201 |
|  9 | 2016-08-06 | C     |        NULL |       201 |
|  7 | 2016-08-09 | C     |         129 |       201 |
| 14 | 2016-07-02 | D     |         172 |      NULL |
| 13 | 2016-09-01 | D     |         165 |       165 |
| 15 | NULL       | D     |         149 |       165 |
| 10 | 2016-06-01 | NULL  |         125 |      NULL |
| 11 | 2016-07-02 | NULL  |         131 |       131 |
| 12 | 2016-08-03 | NULL  |         152 |       131 |
+----+------------+-------+-------------+-----------+
gbase> SELECT *, nth_ value(totalamount, NULL) over (partition by uname
order by dt) as nth_ value from tt;
ERROR 1733 (HY000): (GBA-01EX-700) Gbase general error: argument[2] of
nth_ value is out of range
gbase> SELECT *, nth_ value(totalamount, 0) over (partition by uname
order by dt) as nth_ value from tt;
ERROR 1733 (HY000): (GBA-01EX-700) Gbase general error: argument[2] of
nth_ value is out of range

| 10 | 2016-06-01 | NULL  |         125 | NULL       |
| 11 | 2016-07-02 | NULL  |         131 | const      |
| 12 | 2016-08-03 | NULL  |         152 | const      |
+----+------------+-------+-------------+-----------+
```

5.1.5.8 OLAP function

Precautions for use

GBase 8a MPP Cluster provides rich OLAP functions to assist users in completing complex query statistics. When using these functions, please note the following points:

- The use of aliases is no longer supported within the parentheses of PART BY and ORDER BY in OLAP functions.

  SELECT a AS e ,RANK() OVER(ORDER BY e) FROM t1; -- After AS e, OVER

(ORDER BY e) refers to alias e, which is not supported.

- The integer values in parentheses of the PART BY and ORDER BY in the OLAP function are not used to specify the index of the query result column.

  SELECT a, RANK() OVER(ORDER BY 1) FROM t1; In this query statement, the 1 in the ORDER BY bracket is not used to specify the index referencing the query result column, that is, it does not refer to column a, but is treated as a constant 1.

5.1.5.8.1 GROUP BY class window function

GROUP BY clause syntax

```
GROUP BY { expr | rollup_cube_clause | grouping_sets_clause }, ...
rollup_ cube_ clause:
{ rollup | cube }    ( expr, ...)

grouping_ sets_ clause:
grouping sets ( { rollup_cube_clause | expr }, ...)
```

Table 532 Parameter Description

Parameter Name	Description
rollup_ cube_ clause	Represents a cube function or a roll up function.
expr	The parameters of the cube and roll up functions can be expr, representing a column or a list expression, and can contain one or more exprs.
grouping_ sets_ clause	Represents a grouping_ The sets function, where the parameters of the grouping sets function can be one or more of the cube function, roll up function, column, and list expressions.

explain

- Group by can support one or more functions such as columns, list expressions, cube functions, roll up functions, and grouping sets.

5.1.5.8.1.1 GROUP BY CUBE function

grammar

```
GROUP BY CUBE( (…),(…),…)
```

function

Perform a GROUP BY operation on the n fields or expression combinations in parentheses after CUBE, and finally merge the results together to form a complete subset of the n fields or expressions.

explicate

GROUP BY CUBE (A, B, C)

Firstly, a GROUP BY operation will be performed on (A, B, C), followed by (A, B), (A, C), (A), (B, C), (B), and (C). Then, a GROUP BY operation will be performed on the entire table. Finally, all results will be merged together (equivalent to a UNION ALL operation). If one or more of the n fields or expressions does not appear after GROUP BY in a certain group, NULL will be used to replace the fields or expressions that do not appear.

Example

Tables and data used in the example:

```
DROP TABLE IF EXISTS t3;
CREATE TABLE t3 (color_type varchar(20),color_ count int, in_ date date);
INSERT INTO t3 (color_type,in_date,color_count)
VALUES('black','2010-09-11',18),
('black','2010-10-05',18),('black','2010-10-13',31),
('blue','2010-09-21',23),('blue','2010-09-30',15),
('blue','2010-10-11',62),('red','2010-09-12',41),
('red','2010-10-01',12),('red','2010-10-05',11);
```

Example 1: GROUP BY CUBE (color_type, f_YearMonth)

```
gbase> SELECT color_ type,in_ date,color_ count FROM t3 ORDER BY color_ type,in_ date;
+------------+------------+-------------+
| color_ type | in_ date        | color_ count |
+------------+------------+-------------+
| black           | 2010-09-11 |                  18 |
| black           | 2010-10-05 |                  18 |
| black           | 2010-10-13 |                  31 |
| blue            | 2010-09-21 |                  23 |
| blue            | 2010-09-30 |                  15 |
| blue            | 2010-10-11 |                  62 |
| red             | 2010-09-12 |                  41 |
| red             | 2010-10-01 |                  12 |
| red             | 2010-10-05 |                  11 |
+------------+------------+-------------+
9 rows in set

gbase> SELECT NVL(color_type,'') as color_ type_ Show, NVL (DECODE (color_type, NULL, f_YearMonth | |'Total ', NVL (f_YearMonth, color_type | |'Subtotal')), 'Total') AS f_ YearMonth_ show,SUM(color_count)
 FROM (SELECT color_type,DATE_FORMAT(in_date, '%Y-%m') as f_
```

```
 YearMonth,color_ count FROM t3) t GROUP BY CUBE(color_type,f_Y
earMonth) ORDER BY color_ type,f_ YearMonth;
+-----------------+------------------+------------------+
| color_ type_ show | f_ YearMonth_ show | SUM(color_count) |
+-----------------+------------------+------------------+
| black                | 2010-09              |                    18 |
| black                | 2010-10              |                    49 |
|Black | Black Subtotal | 67|
| blue                 | 2010-09              |                    38 |
| blue                 | 2010-10              |                    62 |
|Blue | Subtotal of blue | 100|
| red                  | 2010-09              |                    41 |
| red                  | 2010-10              |                    23 |
|Red | Subtotal of red | 64|
|2010-09 Total | 97|
|2010-10 Total | 134|
|Total | 231|
+-----------------+------------------+------------------+
12 rows in set
```

5.1.5.8.1.2 GROUP BY ROLLUP function

grammar

```
GROUP BY ROLLUP( (…),(…),…)
```

function

Perform a GROUP BY operation on the n fields or expression combinations in parentheses after ROLLUP, and finally merge the results together using n, n-1, n-2,..., 1, 0.

explicate

GROUP BY ROLLUP (A, B, C)

1. Firstly, a GROUP BY will be performed on (A, B, C);
2. Then perform a GROUP BY on (A, B);
3. Then (A) perform a GROUP BY;
4. Finally, perform a GROUP BY operation on the entire table, that is, perform a group by operation on fields or expressions that do not appear in the GROUP BY ROLLUP function, and replace those that do not appear with NULL
5. Finally, merge all the results together (equivalent to a UNION ALL operation).

If one or more of n fields or expressions do not appear after GROUP BY in a certain group, use NULL to replace the fields or expressions that do not appear.

```
create database if not exists db1;
drop table if exists t1;
create table t1 (a int,b int,c int);
insert into t1 values (1,3,5),(2,4,6);
gbase> select * from t1 group by rollup(a,b,c);
+------+------+------+
| a    | b    | c    |
+------+------+------+
|    1 |    3 |    5 |①
|    2 |    4 |    6 |①
|    1 |    3 | NULL |②
|    2 |    4 | NULL |②
|    1 | NULL | NULL |③
|    2 | NULL | NULL |③
| NULL | NULL | NULL |④
+------+------+------+
7 rows in set (Elapsed: 00:00:00.08)
gbase> select *,sum(a),max(b),min(c) from t1 group by rollup(a,b,c);
+------+------+------+--------+--------+--------+
| a    | b    | c    | sum(a) | max(b) | min(c) |
+------+------+------+--------+--------+--------+
|    2 |    4 |    6 |      2 |      4 |      6 |①
|    1 |    3 |    5 |      1 |      3 |      5 |①
|    2 |    4 | NULL |      2 |      4 |      6 |②
|    1 |    3 | NULL |      1 |      3 |      5 |②
|    2 | NULL | NULL |      2 |      4 |      6 |③
|    1 | NULL | NULL |      1 |      3 |      5 |③
| NULL | NULL | NULL |      3 |      4 |      5 |④
+------+------+------+--------+--------+--------+
7 rows in set (Elapsed: 00:00:00.09)
```

Usually, this function is used for statistical scenarios such as product details, subtotals, and final totals.

Example

Tables and data used in the example:

```
DROP TABLE IF EXISTS t3;
```

```
CREATE TABLE t3 (color_type varchar(20),color_ count int, in_ date date);
INSERT INTO t3 (color_type,in_date,color_count)
VALUES('black','2010-09-11',18),
('black','2010-10-05',18),('black','2010-10-13',31),
('blue','2010-09-21',23),('blue','2010-09-30',15),
('blue','2010-10-11',62),('red','2010-09-12',41),
('red','2010-10-01',12),('red','2010-10-05',11);
```

Example 1: GROUP BY ROLLUP (color_type, f_YearMonth)

```
gbase> SELECT NVL(color_type,'') as color_ type_ Show, DECODE (NVL (color_type, ''), '', 'Total', NVL (f_YearMonth, color_type | | 'Subtotal') AS f_ YearMonth_ show,SUM(color_count) FROM (SELECT color_type,DATE_FORMAT(in_date, '%Y-%m') as f_ YearMonth,color_ count FROM t3) t GROUP BY ROLLUP(color_type,f_YearMonth) ORDER BY color_ type,f_ YearMonth;
+-----------------+------------------+------------------+
| color_ type_ show | f_ YearMonth_ show | SUM(color_count) |
+-----------------+------------------+------------------+
| black | 2010-09 | 18 |
| black | 2010-10 | 49 |
|Black | Black Subtotal | 67|
| blue | 2010-09 | 38 |
| blue | 2010-10 | 62 |
|Blue | Subtotal of blue | 100|
| red | 2010-09 | 41 |
| red | 2010-10 | 23 |
|Red | Subtotal of red | 64|
|Total | 231|
+-----------------+------------------+------------------+
10 rows in set
```

5.1.5.8.1.3 GROUP BY GROUPING SETS function

grammar

```
GROUP BY GROUPING SETS( (…),(…),…)
```

function

Perform a GROUP BY operation on the n fields or expressions in parentheses after GROUPING SETS, and finally merge the results together.

explicate

GROUP BY GROUPING SETS (A, B, C) (A, B, C represent "(...)" in grammar)

Firstly, perform a GROUP BY on (A), followed by a GROUP BY on (B), followed by a

GROUP BY on (C), and finally merge all the results together (equivalent to a UNION ALL operation). If one or more of n fields or expressions do not appear after a GROUP BY in a certain group, use NULL to replace the fields or expressions that do not appear.

Example

Tables and data used in the example:

```
DROP TABLE IF EXISTS t3;
CREATE TABLE t3 (color_type varchar(20),color_ count int, in_ date date);
INSERT INTO t3 (color_type,in_date,color_count)
VALUES('black','2010-09-11',18),
('black','2010-10-05',18),('black','2010-10-13',31),
('blue','2010-09-21',23),('blue','2010-09-30',15),
('blue','2010-10-11',62),('red','2010-09-12',41),
('red','2010-10-01',12),('red','2010-10-05',11);
```

Example 1: GROUP BY GROUPING SETS (color_type, f_YearMonth)

```
gbase> SELECT color_ type,in_ date,color_ count FROM t3 ORDER BY color_ type,in_ date;
+------------+------------+-------------+
| color_ type | in_ date       | color_ count |
+------------+------------+-------------+
| black          | 2010-09-11 |                18 |
| black          | 2010-10-05 |                18 |
| black          | 2010-10-13 |                31 |
| blue           | 2010-09-21 |                23 |
| blue           | 2010-09-30 |                15 |
| blue           | 2010-10-11 |                62 |
| red            | 2010-09-12 |                41 |
| red            | 2010-10-01 |                12 |
| red            | 2010-10-05 |                11 |
+------------+------------+-------------+
9 rows in set

gbase> SELECT NVL(color_type,'') as color_ type_ Show, DECODE (color_type, NULL, f_YearMonth | |'Total ', NVL (f_YearMonth, color_type | |'Subtotal') AS f_ YearMonth_ show,SUM(color_count) FROM (SELECT color_type,DATE_FORMAT(in_date, '%Y-%m') as f_ YearMonth,color_ count FROM t3) t GROUP BY GROUPING SETS(color_type,f_YearMonth) ORDER BY color_ type,f_ YearMonth;
+-----------------+------------------+------------------+
| color_ type_ show | f_ YearMonth_ show | SUM(color_count) |
+-----------------+------------------+------------------+
|Black | Black Subtotal | 67|
```

```
|Blue | Subtotal of blue | 100|
|Red | Subtotal of red | 64|
|2010-09 Total | 97|
|2010-10 Total | 134|
+-----------------+------------------+------------------+
5 rows in set
```

5.1.5.8.2 Non GROUP BY class window function

5.1.5.8.2.1 RANK OVER function

grammar

```
RANK() OVER([PARTITION BY col_name1,col_name2,…] ORDER BY col_name1 [ASC/DESC], col_name2 [ASC/DESC],…)
```

Function Description

Calculate the relative position of each row returned by the query relative to other rows based on the value of the expression in the ORDER BY clause. The data within the group is sorted according to the ORDER BY clause, and each row is assigned a number to form a sequence that starts at 1 and accumulates from there.

Every time the value of the ORDER BY expression changes, the sequence also increases accordingly. Rows with the same value receive the same numerical number (assuming null is equal).

If two rows obtain the same sort, the subsequent ordinals will jump. For example, if two rows have an ordinal of 1 and there is no ordinal of 2, the sequence will assign a value of 3 to the next row in the group.

Only supported by the Express engine.

In the query statement, the clauses that can use the RANK function are:

- In the SELECT list:

```
SELECT RANK() OVER(PARTITION BY i ORDER BY j) FROM t1 WHERE ...;
```

- In the final ORDER BY clause (by using aliases or positional references of the RANK function elsewhere in the query):

```
SELECT *,RANK() OVER (ORDER BY j) FROM t1 WHERE ... ORDER BY RANK() OVER(ORDER BY i DESC);
```

- In the above two clauses, as parameters of an expression or scalar function:

```
SELECT RANK() OVER (ORDER BY j DESC) + i FROM t1 WHERE ...;
SELECT CONV(RANK() OVER(PARTITION BY i ORDER BY j ASC),10,2)FROM t
```

Use constraints

The following situations cannot be used:

- In the search criteria of the WHERE clause:

```
SELECT ... FROM t1 WHERE RANK() OVER(ORDER BY i) > 3; -- error
```

- As parameters of the aggregation function:

```
SELECT SUM(RANK() OVER(ORDER BY dollars)) FROM t1; -- error
```

- The RANK function must not be used in the HAVING clause:

```
SELECT * FROM t1 GROUP BY i HAVING RANK() OVER(ORDER BY j) <
10; -- error
```

- The RANK function must not be in the GROUP BY LIST:

```
SELECT * FROM t1 GROUP BY RANK() OVER(ORDER BY i); -- error
```

- RANK cannot be nested inside other RANKs:

```
SELECT RANK() OVER(ORDER BY RANK() OVER(ORDER BY i )); -- error
```

- The RANK function must not be in the non query part of DELETE and UPDATE statements:

```
UPDATE t1 SET i = RANK () OVER(ORDER BY j) WHERE ...; -- error
```

- But when used in the query section, it can:

```
UPDATE t1 SET i = i + 1 where j IN (SELECT RANK () OVER(ORDER BY
t2.k) from t2);    -- ok
```

be careful

The 'PART BY' cannot be followed by ASC/DESC, but the 'ORDER BY' can be followed by ASC/DESC.

Example

Example 1: RANK() OVER (PART BY i ORDER BY j desc)

```
gbase> DROP TABLE IF EXISTS t1;
Query OK, 0 rows affected

gbase> CREATE TABLE t1(i int, j int);
Query OK, 0 rows affected
```

```
gbase> INSERT INTO t1 VALUES(2,1),(2,3),(2,3),(2,5),(3,2),(3,2),(3,2),(3,4),(3,1),(3,5);
Query OK, 10 rows affected
Records: 10  Duplicates: 0  Warnings: 0

gbase> SELECT *,RANK() OVER(PARTITION BY i ORDER BY j desc) AS rank FROM t1;
+------+------+------+
| i    | j    | rank |
+------+------+------+
|    2 |    5 |    1 |
|    2 |    3 |    2 |
|    2 |    3 |    2 |
|    2 |    1 |    4 |
|    3 |    5 |    1 |
|    3 |    4 |    2 |
|    3 |    2 |    3 |
|    3 |    2 |    3 |
|    3 |    2 |    3 |
|    3 |    1 |    6 |
+------+------+------+
10 rows in set
```

5.1.5.8.2.2 DENSE_ RANK OVER function

grammar

```
DENSE_ RANK( ) over([PARTITION BY col_name1,col_name2,…] ORDER BY col_name1 [ASC/DESC], col_name2 [ASC/DESC],…)
```

Function Description

The basic function is similar to rank, except that if two rows have the same sort, the subsequent ordinals do not jump. For example, if two rows have an ordinal number of 1, the sequence will assign a value of 2 to the next row in the group.

Only supported by the Express engine.

The usage instructions and constraints are the same as RANK() Over().

Example

Example 1: rank, DENSE_ RANK() OVER (partition by i order by j desc)

```
gbase> DROP TABLE IF EXISTS t1;
Query OK, 0 rows affected
```

```
gbase> CREATE TABLE t1(i int, j int);
Query OK, 0 rows affected

gbase> INSERT INTO t1 VALUES(2,1),(2,3),(2,3),(2,5),(3,2),(3,2),(3,2),(3,4),(3,1),(3,5);
Query OK, 10 rows affected
Records: 10  Duplicates: 0  Warnings: 0

gbase> SELECT *,RANK() OVER(PARTITION BY i ORDER BY j DESC) AS rank,DENSE_ RANK() OVER (partition by i order by j desc) AS dense_ rank FROM t1;
+------+------+------+------------+
| i    | j    | rank | dense_ rank |
+------+------+------+------------+
|    2 |    5 |    1 |          1 |
|    2 |    3 |    2 |          2 |
|    2 |    3 |    2 |          2 |
|    2 |    1 |    4 |          3 |
|    3 |    5 |    1 |          1 |
|    3 |    4 |    2 |          2 |
|    3 |    2 |    3 |          3 |
|    3 |    2 |    3 |          3 |
|    3 |    2 |    3 |          3 |
|    3 |    1 |    6 |          4 |
+------+------+------+------------+
10 rows in set
```

5.1.5.8.2.3 ROW_ NUMBER OVER function

grammar

```
ROW_ NUMBER( ) OVER([PARTITION BY col_name1,col_name2, … ] ORDER BY col_name1 [asc/desc], col_name2 [asc/desc],…)
```

Function Description

The difference with rank is that the same sorting value sequence number will also increase in sequence.

For example, if two rows have the same sorting value, the ordinal is 1, 2.

The usage instructions and constraints are the same as RANK() Over().

Example

Example 1: ROW_ NUMBER() OVER(PARTITION BY i order by j desc)

```
gbase> DROP TABLE IF EXISTS t1;
Query OK, 0 rows affected

gbase> CREATE TABLE t1(i int, j int);
Query OK, 0 rows affected

gbase> INSERT INTO t1 VALUES(2,1),(2,3),(2,3),(2,5),(3,2),(3,2),(3,2),(3,4),(3,1),(3,5);
Query OK, 10 rows affected
Records: 10  Duplicates: 0  Warnings: 0

gbase> SELECT *,RANK() OVER(PARTITION BY i order by j desc) AS rank,DENSE_ RANK() OVER(PARTITION BY i order by j desc) AS dense_ rank ,ROW_ NUMBER() OVER(PARTITION BY i order by j desc) AS row_ number FROM t1;
+------+------+------+------------+------------+
| i    | j    | rank | dense_ rank | row_ number |
+------+------+------+------------+------------+
|    2 |    5 |    1 |          1 |          1 |
|    2 |    3 |    2 |          2 |          2 |
|    2 |    3 |    2 |          2 |          3 |
|    2 |    1 |    4 |          3 |          4 |
|    3 |    5 |    1 |          1 |          1 |
|    3 |    4 |    2 |          2 |          2 |
|    3 |    2 |    3 |          3 |          3 |
|    3 |    2 |    3 |          3 |          4 |
|    3 |    2 |    3 |          3 |          5 |
|    3 |    1 |    6 |          4 |          6 |
+------+------+------+------------+------------+
10 rows in set
```

5.1.5.8.2.4 SUM OVER function

grammar

```
SUM([DISTINCT/ALL] expr) OVER([PARTITION BY …]  [ORDER BY … [ASC/DESC] ])
```

Function Description

Calculate the moving sum of expressions within a group.

Example

Example 1: SUM (k) Over (PART BY i ORDER BY j DESC)

```
gbase> DROP TABLE IF EXISTS t1;
Query OK, 0 rows affected

gbase> CREATE TABLE t1(i int, j int,k int);
Query OK, 0 rows affected

gbase> INSERT INTO t1 VALUES(2,1,4), (2,3,6),(2,3,4),(2,5,8), (3,2,2),(3,2,4), (3,2,2),(3,4,6),(3,1,2),(3,5,8);
Query OK, 10 rows affected
Records: 10  Duplicates: 0  Warnings: 0

gbase> SELECT *,SUM(k) OVER(PARTITION BY i ORDER BY j DESC) AS sum FROM t1;
+------+------+------+------+
| i    | j    | k    | sum  |
+------+------+------+------+
|    2 |    5 |    8 |    8 |
|    2 |    3 |    4 |   18 |
|    2 |    3 |    6 |   18 |
|    2 |    1 |    4 |   22 |
|    3 |    5 |    8 |    8 |
|    3 |    4 |    6 |   14 |
|    3 |    2 |    2 |   22 |
|    3 |    2 |    4 |   22 |
|    3 |    2 |    2 |   22 |
|    3 |    1 |    2 |   24 |
+------+------+------+------+
10 rows in set
```

Use case analysis: Firstly, it will be grouped according to i, arranged in descending order of j within the same group. Starting from the first value of each group, the k value will be accumulated backwards. The same j value, corresponding to the same cumulative sum, will be added to the k value corresponding to the last j value. If encountering different groups, it will be accumulated again from 0.

be careful

The processing method of NULL values is similar to that of the aggregate function sum. If all values are NULL, the result is NULL, otherwise NULL is not accumulated.

Taking i values of 2, 2, 2, 2, j values of 5, 3, 3, 1, K values of 8, 4, 6, 4, and sum values of 8, 18, 18, and 22 as examples, when i=2, j=5, k=8, sum=8, i=2, j=3, k=4, and i=2, j=3,

k=6, because j values are the same, sum values are the same. The calculation process is sum=8+4+6=18.

Example 2: SUM (distinct k) Over (PART BY i)

```
gbase> SELECT *,SUM(distinct k) OVER(PARTITION BY i) AS sum FROM t1;
+------+------+------+------+
| i    | j    | k    | sum  |
+------+------+------+------+
|    2 |    3 |    6 |   18 |
|    2 |    3 |    4 |   18 |
|    2 |    5 |    8 |   18 |
|    2 |    1 |    4 |   18 |
|    3 |    2 |    2 |   20 |
|    3 |    2 |    4 |   20 |
|    3 |    2 |    2 |   20 |
|    3 |    4 |    6 |   20 |
|    3 |    1 |    2 |   20 |
|    3 |    5 |    8 |   20 |
+------+------+------+------+
10 rows in set
```

Use case analysis: Firstly, group based on i. Since there is no ORDER BY part, the cumulative sum within the same group is equal. Accumulate the non repeating k values within the same group. If encountering different groups, start from 0 again.

Taking i values of 2, 2, 2, 2, j values of 5, 3, 3, 1, k values of 8, 4, 6, 4, and sum values of 18, 18, 18, and 18 as examples, because in these four sets of values, different k values are 6, 4, and 8, so sum=6+4+8=18.

5.1.5.8.2.5 AVG Over function

grammar

```
AVG([DISTINCT/ALL] expr) OVER([PARTITION BY …] [ORDER BY … [ASC/DESC] ])
```

Function Description

Calculate the moving average of expressions within a group.

Example

Example 1: AVG (k) Over (PART BY i ORDER BY j DESC)

```
gbase> DROP TABLE IF EXISTS t1;
```

```
Query OK, 0 rows affected, 1 warning

gbase> CREATE TABLE t1(i int, j int,k int);
Query OK, 0 rows affected

gbase> INSERT INTO t1 VALUES(2,1,4),(2,3,6),(2,3,4),(2,5,8),(3,2,2), (3,2,
4),(3,2,2), (3,4,6),(3,1,2),(3,5,8);
Query OK, 10 rows affected
Records: 10  Duplicates: 0  Warnings: 0

gbase> SELECT *,AVG(k) OVER(PARTITION BY i ORDER BY j DES
C) AS avg FROM t1;
+------+------+------+--------+
| i    | j    | k    | avg    |
+------+------+------+--------+
|    2 |    5 |    8 | 8.0000 |
|    2 |    3 |    4 | 6.0000 |
|    2 |    3 |    6 | 6.0000 |
|    2 |    1 |    4 | 5.5000 |
|    3 |    5 |    8 | 8.0000 |
|    3 |    4 |    6 | 7.0000 |
|    3 |    2 |    2 | 4.4000 |
|    3 |    2 |    4 | 4.4000 |
|    3 |    2 |    2 | 4.4000 |
|    3 |    1 |    2 | 4.0000 |
+------+------+------+--------+
10 rows in set
```

Use case analysis: Firstly, it will be grouped according to i and arranged in descending order of j within the same group. Starting from the first value of each group, the k value will be accumulated backwards. At the same time, the value of count (k) will be recorded. If the j value is the same, the corresponding cumulative sum and count value will be the same, and they will be calculated to the k value corresponding to the last j value. If different groups are encountered, they will be re accumulated from 0. Finally, the cumulative sum will be divided by the count value to obtain the final avg value.

be careful

The processing method of NULL values is similar to that of the aggregate function avg. If all values are NULL, the result is NULL. Otherwise, NULL is not accumulated and is not counted in the count.

Taking i values of 2, 2, 2, 2, j values of 5, 3, 3, 1, k values of 8, 4, 6, 4, and sum values of 8, 6, 6, and 5.5 as examples, when i=2, j=5, k=8, avg=8, i=2, j=3, k=4, and i=2, j=3, k=6, because j values are the same, avg values are the same. The calculation process is avg=(8+4+6)/3=6, i=2, j=1, k=4, avg=5.5, and avg=(8+4+6+4)/4=5.5.

Example 2: AVG (DISTINCT k) Over (PART BY i)

```
gbase> SELECT *,AVG(DISTINCT k) OVER(PARTITION BY i) AS av
g FROM t1;
+------+------+------+--------+
| i    | j    | k    | avg    |
+------+------+------+--------+
|    2 |    3 |    6 | 6.0000 |
|    2 |    3 |    4 | 6.0000 |
|    2 |    5 |    8 | 6.0000 |
|    2 |    1 |    4 | 6.0000 |
|    3 |    2 |    2 | 5.0000 |
|    3 |    2 |    4 | 5.0000 |
|    3 |    2 |    2 | 5.0000 |
|    3 |    4 |    6 | 5.0000 |
|    3 |    1 |    2 | 5.0000 |
|    3 |    5 |    8 | 5.0000 |
+------+------+------+--------+
10 rows in set
```

Use case analysis: Firstly, group based on i. Since there is no ORDER BY part, the cumulative sum and COUNT values within the same group are equal. Accumulate the k values within the same group and calculate the COUNT value. If encountering different groups, restart from 0.

Taking i values of 2, 2, 2, 2, j values of 5, 3, 3, 1, k values of 8, 4, 6, 4, and avg values of 6, 6, 6, and 6 as examples, because among these four sets of values, different k values are 6, 4, and 8, avg=(6+4+8)/3=6.

5.1.5.8.2.6 COUNT OVER function

grammar

```
COUNT(*/[DISTINCT] col ) OVER([PARTITION BY col_name1,col_name2,…]
[ORDER BY col_name1 [ASC/DESC], col_name2 [ASC/DESC],…] )
```

Function Description

This function is used to calculate the number of records in a group. If it is COUNT (*), the NULL value does not need to be considered. Otherwise, records with a parameter of NULL are not included. If it contains DISTINCT, a deduplication operation needs to be

performed.

Example

Example 1: Example of the COUNT OVER function.

```
gbase> DROP TABLE IF EXISTS t2;
Query OK, 0 rows affected

gbase> CREATE TABLE t2(i int,j int,k int);
Query OK, 0 rows affected

gbase> INSERT INTO t2 VALUES(2,1,4),(2,3,6),(2,3,4),(2,5,8),(3,2,2),(3,2,
4),(3,2,2), (3,4,6),(3,1,2),(3,5,8);
Query OK, 10 rows affected
Records: 10  Duplicates: 0  Warnings: 0

gbase> SELECT *,COUNT(k) OVER(PARTITION BY i ORDER BY j
DESC) AS sum FROM t2;
+------+------+------+-----+
| i    | j    | k    | sum |
+------+------+------+-----+
|    2 |    5 |    8 |   1 |
|    2 |    3 |    4 |   3 |
|    2 |    3 |    6 |   3 |
|    2 |    1 |    4 |   4 |
|    3 |    5 |    8 |   1 |
|    3 |    4 |    6 |   2 |
|    3 |    2 |    2 |   5 |
|    3 |    2 |    4 |   5 |
|    3 |    2 |    2 |   5 |
|    3 |    1 |    2 |   6 |
+------+------+------+-----+
10 rows in set

gbase> SELECT *,COUNT(DISTINCT k) OVER(PARTITION BY i) AS
 sum FROM t2;
+------+------+------+-----+
| i    | j    | k    | sum |
+------+------+------+-----+
|    2 |    3 |    6 |   3 |
|    2 |    3 |    4 |   3 |
|    2 |    5 |    8 |   3 |
|    2 |    1 |    4 |   3 |
|    3 |    2 |    2 |   4 |
```

```
|      3 |      2 |      4 |    4 |
|      3 |      2 |      2 |    4 |
|      3 |      4 |      6 |    4 |
|      3 |      1 |      2 |    4 |
|      3 |      5 |      8 |    4 |
+------+------+------+-----+
10 rows in set
```

5.1.5.8.2.7 How to use the clause 'range/row between' when opening a window

Grammar:

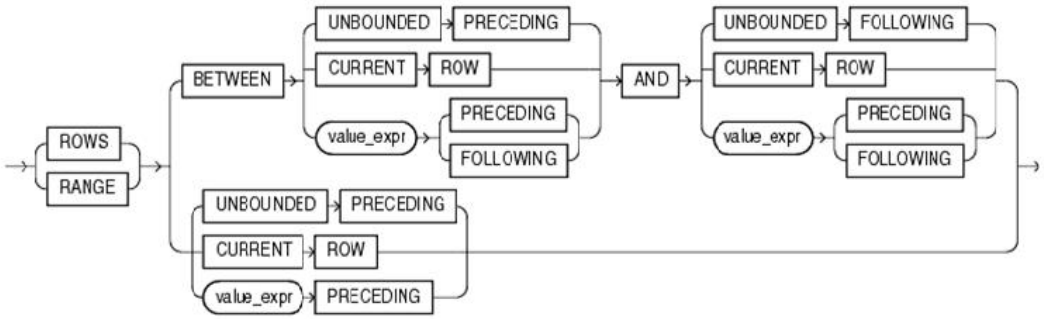
func(expr)over([partition_clause][order_by_clause][windowing_clause]

Window clause in OLAP function: [windowing_clause] clause

The scope of use and specific syntax are as follows:

Support for windowing_ The OLAP func functions of clause include: sum, avg, count, max, min, and do not support the distinct keyword.

windowing_ The syntax diagram of 'close' is as follows



Keyword Description:

Rows specifies the range of rows	Range specifies the range of values
Unbounded	Advance
Following down, adding, delaying	Current row

Example:

Table structure: create table t (a int, b int, c int, d datetime);

Example: Group data in column a, sort data in column b or column d, and output the sum of column c and sum (c) within the specified range of window clauses within the same group

Control parameters that support clauses need to be opened in advance:

set _ t_ gbase_ new_ window_ function_ support=1

Example 1: Window that specifies the range of rows, from the previous row of the current row to the current row.

gbase> select a,b,c,sum(c)over(partition by a order by b,c rows between 1 preceding and current row) as 'sum(c)' from t;

```
+------+------+------+--------+
| a    | b    | c    | sum(c) |
+------+------+------+--------+
|    2 |    1 |    1 |      1 |
|    2 |    2 |    1 |      2 |
|    1 |    1 |    2 |      2 |
|    1 |    2 |    4 |      6 |
|    1 |    3 |    5 |      9 |
+------+------+------+--------+
```

5 rows in set (Elapsed: 00:00:00.21)

Example 2: Window that specifies the range of rows, from the previous row of the current row to the next row of the current row.

gbase> select a,b,c,sum(c)over(partition by a order by b,c rows between 1 preceding and 1 following) as 'sum(c)' from t;

```
+------+------+------+--------+
| a    | b    | c    | sum(c) |
+------+------+------+--------+
|    2 |    1 |    1 |      2 |
|    2 |    2 |    1 |      2 |
|    1 |    1 |    2 |      6 |
|    1 |    2 |    4 |     11 |
|    1 |    3 |    5 |      9 |
+------+------+------+--------+
```

5 rows in set (Elapsed: 00:00:00.07)

Example 3: A window with a specified value range, where the value of the current row value column (column b) is x, and the window is within the closed range of [(x-2), x] for column b, outputs the cumulative sum of column c in the same group within that range.

gbase> select a,b,c,sum(c)over(partition by a order by b range between 2 preceding and current row) as 'sum(c)' from t;

```
+------+------+------+--------+
| a    | b    | c    | sum(c) |
+------+------+------+--------+
|    2 |    1 |    1 |      1 |
|    2 |    2 |    1 |      2 |
|    1 |    1 |    2 |      2 |
```

```
|      1 |      2 |      4 |          6 |
|      1 |      3 |      5 |         11 |
+------+------+------+--------+
```

5 rows in set (Elapsed: 00:00:00.10)

Example 4: A window with a specified value range, where the date column d in the current row has a date value of x, and the window is a range where the date value of column d falls within the closed range of [the first 2 days of x, the last 3 days of x]. Output the cumulative sum of column c in the same group within this range.

gbase> select a,d,c,sum(c)over(partition by a order by d range between 2 preceding and 3 following) as 'sum(c)' from t;

```
+------+---------------------+------+--------+
| a    | d                   | c    | sum(c) |
+------+---------------------+------+--------+
|    2 | 2010-10-10 10:00:00 |    1 |      2 |
|    2 | 2010-10-10 10:01:00 |    1 |      2 |
|    1 | 2010-10-08 10:00:00 |    2 |      6 |
|    1 | 2010-10-11 06:00:00 |    4 |      9 |
|    1 | 2010-10-12 10:00:00 |    5 |      9 |
+------+---------------------+------+--------+
```

5 rows in set (Elapsed: 00:00:00.36)

Example 5: A window with a specified value range, where the date column d in the current row has a date value of x, and the window is a range where the date value of column d falls within the closed range of [beginning of row, last 3 minutes of x]. Output the cumulative sum of column c in the same group within this range.

gbase> select a,d,c,sum(c)over(partition by a order by d range between unbounded preceding and interval 3 minute following) as 'sum(c)' from t;

```
+------+---------------------+------+--------+
| a    | d                   | c    | sum(c) |
+------+---------------------+------+--------+
|    1 | 2010-10-08 10:00:00 |    2 |      2 |
|    1 | 2010-10-11 06:00:00 |    4 |      6 |
|    1 | 2010-10-12 10:00:00 |    5 |     11 |
|    2 | 2010-10-10 10:00:00 |    1 |      2 |
|    2 | 2010-10-10 10:01:00 |    1 |      2 |
+------+---------------------+------+--------+
```

5 rows in set (Elapsed: 00:00:00.07)

Use constraints:

1. _ t_ gbase_ new_ window_ function_ The support parameter controls support for

windowing_ The parameter value of the clause clause defaults to 0 and does not support windowing_ Clause clause, supports windowing when set to 1_ Clause clause.

set _ t_ gbase_ new_ window_ function_ support=1

2. Windows not supported_ In the case of the clause clause, the window opening method is fixed, and the range is:

Range between unbounded preceding and current row

3. If the Partition clause is empty, there is no grouping, indicating that all data is in one group; If the order by clause is empty, all data in each group is not sorted, meaning that a group is a "sub window".

4. between bound1 and bound2

Bound1 defines the starting position of the window, and Bound2 defines the ending position of the window.

When a single boundary is used, it is the definition of the starting position, and the ending position defaults to the current row, such as:

```
select a,b,c,sum(c)over(partition by a order by b range 2 preceding) as 'sum(c)' from t;
```

5. unbounded preceding

Indicates the starting position, which is the beginning of the line in the current group, and cannot appear in bound2.

6. unbounded following

Indicates the end position, which is the end of the line in the current group and cannot appear in bound1.

7. current row

Indicates that the starting or ending position is the current line.

8. In the following cases, there can be multiple expressions after the order by keyword:

Range between unbounded preceding and current row

Range between unbounded preceding and unbounded following

Range between current row and current row

Range between current row and unbounded following

9. value_ expr preceding | value_ expr following

Starting at value_ When expr following, the end position should be value_ expr following;

End at value_ When expr preceding, the starting position should be value_ expr preceding。

10. Rows keyword followed by value_ During expr, the interval clause is not supported

value_ When expr is num, it identifies the offset of the row and is a numerical constant, a positive integer (rounded). Cannot be negative, numerical function, or table column.

11. Range keyword followed by value_ During expr, the interval clause is supported, and there can only be one expression after order by:

value_ When expr is num, the expression after order by can only be of numerical or date (time) type;

value_ When expr is an interval clause, the expression after order by can only be of type date (time);

value_ When expr is num, it identifies the offset of the row and is a numerical constant, a positive integer (rounded). Cannot be negative, numerical function, or table column.

12. The range keyword, order by, followed by a unique expression, is used to calculate the window range:

When there is a null value in the sorting and it is at the beginning of a row, unless it is unbounded preceding, the row is not included in the window range;

When sorting with null values and at the end of a row, unless unbounded following, the row is not included in the window range;

The current behavior is a null value, unless unbounded preceding/scrolling, the window range is only for that row.

13. The supported range of date (time) types: date, datetime, and timestamp.

5.1.5.8.2.8 LEAD/LAG OVER function

grammar

```
LEAD/LAG(expr [,offset [,DEFAULT]] ) OVER([PARTITION BY
```

```
col_name1,col_name2,…] ORDER BY col_name1 [ASC/DESC], col_name2 [ASC/DESC],…)
```

Function Description

Supports OLAP functions LEAD() Over(), LAG() Over(). These two functions are offset functions that are used to find N values under or N values above the same field and store them as new columns in the table. LEAD is offset downwards and LAG is offset upwards (N is a non negative integer).

instructions

- Expr: This parameter is an expression for finding the offset.
- Offset: This parameter is the offset and can be omitted. The default value is 1.
- Default: This parameter is the default value and can be omitted. The default value is NULL.
- The rules in these two functions over are the same as those in the RANK class OLAP function.
- This function can return any supported data type.

Use constraints and limitations

Restrictions on parameters:

Arguments 2 and 3 must be constants or constant expressions.

Data type conversion

If the data type of the first parameter is different from that of the third parameter, the third parameter will be implicitly converted based on the data type of the first parameter.

Table -533 Conversion of the first and third parameters

First parameter	Third parameter	After conversion
VARCHAR(2)	VARCHAR(20)	VARCHAR(20)
INT	DECIMAL	INT (rounding)
DECIMAL	INT	DECIMAL
DATE	DATETIME/TIMESTAMP	DATE
DATETIME	DATE	DATETIME
INT	INT (non numerical)	0
INT	DATE	INT OR BIGINT
INT	DATE	VARCHAR
DATE	VARCHAR	Convert INT values that match the date format to the corresponding date, and convert those that do not match to NULL and report warnings

First parameter	Third parameter	After conversion
DATE	INT	Convert INT values that match the date format to the corresponding date, and convert those that do not match to NULL and report warnings
INT	DATE	0

Example

Example 1: LEAD (result, 1, NULL) OVER (PART BY result ORDER BY area DESC)

```
gbase> DROP TABLE IF EXISTS t_ olap;
Query OK, 0 rows affected

gbase> CREATE TABLE t_ olap(result int, area varchar(200));
Query OK, 0 rows affected

gbase> INSERT INTO t_ Olap Values (550, 'Tianjin'), (600, 'Hubei'), (670, 'Hubei');
Query OK, 3 rows affected
Records: 3  Duplicates: 0  Warnings: 0

gbase> INSERT INTO t_ Olap Values (448, 'Tianjin'), (490, 'Beijing'), (598, 'Tianjin'), ('700 ',' Hubei ');
Query OK, 4 rows affected
Records: 4  Duplicates: 0  Warnings: 0

gbase> INSERT INTO t_ Olap Values (528, 'Tianjin'), (446, 'Beijing'), (568, 'Tianjin'), ('682 ',' Hubei ');
Query OK, 4 rows affected
Records: 4  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t_ olap GROUP BY area,result ORDER BY area,result;
+--------+--------+
| result | area   |
+--------+--------+
|446 | Beijing|
|490 | Beijing|
|448 | Tianjin|
|528 | Tianjin|
|550 | Tianjin|
|568 | Tianjin|
|598 | Tianjin|
```

```
|600 | Hubei|
|670 | Hubei|
|682 | Hubei|
|700 | Hubei|
+--------+--------+
11 rows in set

gbase> SELECT *,LEAD(result, 1) OVER(PARTITION BY area ORDER BY area DESC) AS LEAD FROM t_ olap;
+--------+--------+------+
| result | area     | LEAD |
+--------+--------+------+
|550 | Tianjin | 568|
|568 | Tianjin | 528|
|528 | Tianjin | 448|
|448 | Tianjin | 598|
|598 | Tianjin | NULL|
|670 | Hubei | 600|
|600 | Hubei | 682|
|682 | Hubei | 700|
|700 | Hubei | NULL|
|446 | Beijing | 490|
|490 | Beijing | NULL|
+--------+--------+------+
11 rows in set (Elapsed: 00:00:00.19)
```

5.1.5.8.2.9 PERCENT_ RANK() function

grammar

```
PERCENT_ RANK() OVER([PARTITION BY col_name1,col_name2,…]
ORDER BY col_name1 [ASC/DESC], col_name2 [ASC/DESC],…)
```

Function Description

The calculation is defined by the ORDER BY clause and refers to the decimal position of a row relative to other rows in the returned query. It returns a decimal value between 0 and 1.

The usage scenario and limitations of this function are identical to the RANK() function.

5.1.5.8.2.10 GROUPING function

grammar

```
GROUPING (expr)
```

Parameter Description

The set expr must be a group by field, which is in the parameter list of group by roll up/cube/routing sets.

function

GROUPING is used to distinguish between NULL values in data and NULL values returned by GROUP BY class function (ROLLUP, CUBE, GROUPING SETS). The NULL returned as a result of a ROLLUP, CUBE, or GROUPING SETS operation is a special application of NULL. It serves as a placeholder for columns within the result set, representing the whole. GROUPING indicates whether the expressions in the GROUP BY list participate in grouping, returning 1 indicates not participating in grouping, and returning 0 indicates participating in grouping. For a regular GROUP BY expression, GROUPING returns 0.

Example

```
gbase> create table t1(i int,v varchar(10));
Query OK, 0 rows affected (Elapsed: 00:00:00.03)
gbase> insert into t1 values (2,'a'),(2,'b');
Query OK, 2 rows affected (Elapsed: 00:00:00.04)
Records: 2  Duplicates: 0  Warnings: 0
gbase> SELECT * from t1;
+------+------+
| i    | v    |
+------+------+
|    2 | a    |
|    2 | b    |
+------+------+
2 rows in set
gbase> SELECT i,v,grouping(i),grouping(v) from t1 group by rollup(i,v);
+------+------+-------------+-------------+
| i    | v    | grouping(i) | grouping(v) |
+------+------+-------------+-------------+
|    2 | a    |           0 |           0 |
|    2 | b    |           0 |           0 |
|    2 | NULL |           0 |           1 |
| NULL | NULL |           1 |           1 |
+------+------+-------------+-------------+
4 rows in set
gbase> SELECT i,v,grouping(i),grouping(v) from t1 group by grouping sets(i,v);
+------+------+-------------+-------------+
| i    | v    | grouping(i) | grouping(v) |
+------+------+-------------+-------------+
| NULL | a    |           1 |           0 |
| NULL | b    |           1 |           0 |
```

```
|       2 | NULL |                  0 |                  1 |
+------+------+-------------+-------------+
3 rows in set
gbase> SELECT i,v,grouping(i),grouping(v) from t1 group by rollup(i,v) order by grouping(v) desc;
+------+------+-------------+-------------+
| i       | v       | grouping(i) | grouping(v) |
+------+------+-------------+-------------+
|       2 | NULL |                  0 |                  1 |
| NULL | NULL |                   1 |                   1 |
|       2 | b       |                  0 |                  0 |
|       2 | a       |                  0 |                  0 |
+------+------+-------------+-------------+
4 rows in set
gbase> SELECT i,v,grouping(i),grouping(v) from t1 group by rollup(i,v) having grouping(v)>0;
+------+------+-------------+-------------+
| i       | v       | grouping(i) | grouping(v) |
+------+------+-------------+-------------+
|       2 | NULL |                  0 |                  1 |
| NULL | NULL |                   1 |                   1 |
+------+------+-------------+-------------+
2 rows in set
```

5.1.5.8.2.11 PERCENTILE_ CONT function

grammar

```
percentile_ cont(n)within group (order by expr [desc|asc])
over(partition_clause)
```

Table -597 Parameter Description

Parameter Name	explain
n	A numerical constant between 0 and 1 is a percentage expressed in decimal form, rounded to 64 decimal places.
within group (order by *expr* [desc|asc])	The expression specified for sorting and calculating percentages only supports numerical and date types, does not support character types, and does not support multiple columns.
partition_ clause	Set optional parameters for the record range of each group in the over clause.

function

percentile_ The cont function returns a data value that corresponds to the percentage of the input distribution. The type of the function's return value is related to the order by column type. If the column type is numeric, it returns double, and if it is date, it returns date. percentile_ The cont function returns a data value that corresponds to the percentage of the input distribution. The type of the function's return value is related to the order by column type. If the column type is numeric, it returns double, and if it is date, it returns date.

be careful

The columns used for sorting and calculating percentages only support numerical and date types, do not support character types, and do not support multiple columns.

Example

Example 1: Positive sorting

```
gbase> select * from emp;
+------+--------+-----------+------+--------+
| id   | name   | job       | sal  | deptno |
+------+--------+-----------+------+--------+
| 7369 | SMITH  | CLERK     |  800 |     20 |
| 7499 | ALLEN  | SALESMAN  | 1600 |     30 |
| 7521 | WARD   | SALESMAN  | 1250 |     30 |
| 7566 | JONES  | MANAGER   | 2975 |     20 |
| 7654 | MARTIN | SALESMAN  | 1250 |     30 |
| 7698 | BLAKE  | MANAGER   | 2850 |     30 |
| 7782 | CLARK  | MANAGER   | 2450 |     10 |
| 7788 | SCOTT  | ANALYST   | 3000 |     20 |
| 7839 | KING   | PRESIDENT | 5000 |     10 |
| 7844 | TURNER | SALESMAN  | 1500 |     30 |
| 7876 | ADAMS  | CLERK     | 1100 |     20 |
| 7900 | JAMES  | CLERK     |  950 |     30 |
| 7902 | FORD   | ANALYST   | 3000 |     20 |
| 7934 | MILLER | CLERK     | 1300 |     10 |
+------+--------+-----------+------+--------+
14 rows in set (Elapsed: 00:00:00.45)

select name,sal,deptno,cume_ dist() over(order by sal) as pdist , percentile_ cont(0.25) within group(order by sal) over(partition by deptno) as pdisc from emp;
```

```
+--------+------+--------+--------------------+-------+
| name   | sal  | deptno | pdist              | pdisc |
+--------+------+--------+--------------------+-------+
| SMITH  |  800 |     20 | 0.0714285714285714 |  1100 |
| ADAMS  | 1100 |     20 |  0.214285714285714 |  1100 |
| JONES  | 2975 |     20 |  0.785714285714286 |  1100 |
| FORD   | 3000 |     20 |  0.928571428571429 |  1100 |
| SCOTT  | 3000 |     20 |  0.928571428571429 |  1100 |
| MILLER | 1300 |     10 |  0.428571428571429 |  1875 |
| CLARK  | 2450 |     10 |  0.642857142857143 |  1875 |
| KING   | 5000 |     10 |                  1 |  1875 |
| JAMES  |  950 |     30 |  0.142857142857143 |  1250 |
| WARD   | 1250 |     30 |  0.357142857142857 |  1250 |
| MARTIN | 1250 |     30 |  0.357142857142857 |  1250 |
| TURNER | 1500 |     30 |                0.5 |  1250 |
| ALLEN  | 1600 |     30 |  0.571428571428571 |  1250 |
| BLAKE  | 2850 |     30 |  0.714285714285714 |  1250 |
+--------+------+--------+--------------------+-------+
14 rows in set (Elapsed: 00:00:00.25)
```

Example 2: Reverse Sort

```
gbase> select * from emp;
+------+--------+-----------+------+--------+
| id   | name   | job       | sal  | deptno |
+------+--------+-----------+------+--------+
| 7369 | SMITH  | CLERK     |  800 |     20 |
| 7499 | ALLEN  | SALESMAN  | 1600 |     30 |
| 7521 | WARD   | SALESMAN  | 1250 |     30 |
| 7566 | JONES  | MANAGER   | 2975 |     20 |
| 7654 | MARTIN | SALESMAN  | 1250 |     30 |
| 7698 | BLAKE  | MANAGER   | 2850 |     30 |
| 7782 | CLARK  | MANAGER   | 2450 |     10 |
| 7788 | SCOTT  | ANALYST   | 3000 |     20 |
| 7839 | KING   | PRESIDENT | 5000 |     10 |
| 7844 | TURNER | SALESMAN  | 1500 |     30 |
| 7876 | ADAMS  | CLERK     | 1100 |     20 |
| 7900 | JAMES  | CLERK     |  950 |     30 |
| 7902 | FORD   | ANALYST   | 3000 |     20 |
| 7934 | MILLER | CLERK     | 1300 |     10 |
+------+--------+-----------+------+--------+
14 rows in set (Elapsed: 00:00:00.45)

select name,sal,deptno,cume_ dist() over(order by sal desc) as pdist ,
```

```
percentile_ cont(0.25) within group(order by sal desc) over(partition by deptno) as pdisc from emp;

+--------+------+--------+--------------------+-------+
| name   | sal  | deptno | pdist              | pdisc |
+--------+------+--------+--------------------+-------+
| KING   | 5000 |     10 | 0.0714285714285714 |  3725 |
| CLARK  | 2450 |     10 |  0.428571428571429 |  3725 |
| MILLER | 1300 |     10 |  0.642857142857143 |  3725 |
| BLAKE  | 2850 |     30 |  0.357142857142857 |  1575 |
| ALLEN  | 1600 |     30 |                0.5 |  1575 |
| TURNER | 1500 |     30 |  0.571428571428571 |  1575 |
| WARD   | 1250 |     30 |  0.785714285714286 |  1575 |
| MARTIN | 1250 |     30 |  0.785714285714286 |  1575 |
| JAMES  |  950 |     30 |  0.928571428571429 |  1575 |
| SCOTT  | 3000 |     20 |  0.214285714285714 |  3000 |
| FORD   | 3000 |     20 |  0.214285714285714 |  3000 |
| JONES  | 2975 |     20 |  0.285714285714286 |  3000 |
| ADAMS  | 1100 |     20 |  0.857142857142857 |  3000 |
| SMITH  |  800 |     20 |                  1 |  3000 |
+--------+------+--------+--------------------+-------+
14 rows in set (Elapsed: 00:00:00.22)
```

5.1.5.8.2.12 PERCENTILE_ DISC function

grammar

```
percentile_ dics(n)within group (order by expr [desc|asc])
over(partition_clause)
```

Table -597 Parameter Description

Parameter Name	explain
n	A numerical constant between 0 and 1 is a percentage expressed in decimal form, rounded to 64 decimal places.
within group (order by *expr* [desc|asc])	The expression specified for sorting and calculating percentages only supports numerical and date types, does not support character types, and does not support multiple columns.
partition_ clause	Set optional parameters for the record range of each group in the over clause.

function

percentile_ The disc function returns a data value corresponding to the percentage of the input distribution. The type of the function's return value is related to the order by column type. If the column type is numeric, it returns double, and if it is date, it returns date.

be careful

The columns used for sorting and calculating percentages only support numerical and date types, do not support character types, and do not support multiple columns.

Example

Example 1: Positive sorting

```
gbase> select * from emp;
+------+--------+-----------+------+--------+
| id   | name   | job       | sal  | deptno |
+------+--------+-----------+------+--------+
| 7369 | SMITH  | CLERK     |  800 |     20 |
| 7499 | ALLEN  | SALESMAN  | 1600 |     30 |
| 7521 | WARD   | SALESMAN  | 1250 |     30 |
| 7566 | JONES  | MANAGER   | 2975 |     20 |
| 7654 | MARTIN | SALESMAN  | 1250 |     30 |
| 7698 | BLAKE  | MANAGER   | 2850 |     30 |
| 7782 | CLARK  | MANAGER   | 2450 |     10 |
| 7788 | SCOTT  | ANALYST   | 3000 |     20 |
| 7839 | KING   | PRESIDENT | 5000 |     10 |
| 7844 | TURNER | SALESMAN  | 1500 |     30 |
| 7876 | ADAMS  | CLERK     | 1100 |     20 |
| 7900 | JAMES  | CLERK     |  950 |     30 |
| 7902 | FORD   | ANALYST   | 3000 |     20 |
| 7934 | MILLER | CLERK     | 1300 |     10 |
+------+--------+-----------+------+--------+
14 rows in set (Elapsed: 00:00:00.45)

select name,sal,deptno,cume_ dist() over(order by sal) as pdist , percentile_ disc(0.25) within group(order by sal) over(partition by deptno) as pdisc from emp;

+--------+------+--------+---------------------+-------+
| name   | sal  | deptno | pdist               | pdisc |
+--------+------+--------+---------------------+-------+
```

```
| SMITH   |  800 |        20 | 0.0714285714285714 |   1100 |
| ADAMS   | 1100 |        20 |   0.214285714285714 |   1100 |
| JONES   | 2975 |        20 |   0.785714285714286 |   1100 |
| FORD    | 3000 |        20 |   0.928571428571429 |   1100 |
| SCOTT   | 3000 |        20 |   0.928571428571429 |   1100 |
| MILLER | 1300 |        10 |   0.428571428571429 |   1300 |
| CLARK   | 2450 |        10 |   0.642857142857143 |   1300 |
| KING    | 5000 |        10 |                   1 |   1300 |
| JAMES   |  950 |        30 |   0.142857142857143 |   1250 |
| WARD    | 1250 |        30 |   0.357142857142857 |   1250 |
| MARTIN | 1250 |        30 |   0.357142857142857 |   1250 |
| TURNER | 1500 |        30 |                 0.5 |   1250 |
| ALLEN   | 1600 |        30 |   0.571428571428571 |   1250 |
| BLAKE   | 2850 |        30 |   0.714285714285714 |   1250 |
+--------+------+---------+---------------------+--------+
14 rows in set (Elapsed: 00:00:00.26)
```

Example 2: Reverse Sort

```
gbase> select * from emp;
+------+--------+-----------+------+--------+
| id   | name   | job       | sal  | deptno |
+------+--------+-----------+------+--------+
| 7369 | SMITH  | CLERK     |  800 |     20 |
| 7499 | ALLEN  | SALESMAN  | 1600 |     30 |
| 7521 | WARD   | SALESMAN  | 1250 |     30 |
| 7566 | JONES  | MANAGER   | 2975 |     20 |
| 7654 | MARTIN | SALESMAN  | 1250 |     30 |
| 7698 | BLAKE  | MANAGER   | 2850 |     30 |
| 7782 | CLARK  | MANAGER   | 2450 |     10 |
| 7788 | SCOTT  | ANALYST   | 3000 |     20 |
| 7839 | KING   | PRESIDENT | 5000 |     10 |
| 7844 | TURNER | SALESMAN  | 1500 |     30 |
| 7876 | ADAMS  | CLERK     | 1100 |     20 |
| 7900 | JAMES  | CLERK     |  950 |     30 |
| 7902 | FORD   | ANALYST   | 3000 |     20 |
| 7934 | MILLER | CLERK     | 1300 |     10 |
+------+--------+-----------+------+--------+
14 rows in set (Elapsed: 00:00:00.45)

select name,sal,deptno,cume_ dist() over(order by sal desc) as pdist , percentile_ disc(0.25) within group(order by sal desc) over(partition by deptno) as pdisc from emp;

+--------+------+--------+---------------------+--------+
```

```
| name   | sal  | deptno | pdist                  | pdisc |
+--------+------+--------+--------------------+-------+
| KING   | 5000 |     10 | 0.0714285714285714 |  5000 |
| CLARK  | 2450 |     10 |  0.428571428571429 |  5000 |
| MILLER | 1300 |     10 |  0.642857142857143 |  5000 |
| BLAKE  | 2850 |     30 |  0.357142857142857 |  1600 |
| ALLEN  | 1600 |     30 |                0.5 |  1600 |
| TURNER | 1500 |     30 |  0.571428571428571 |  1600 |
| WARD   | 1250 |     30 |  0.785714285714286 |  1600 |
| MARTIN | 1250 |     30 |  0.785714285714286 |  1600 |
| JAMES  |  950 |     30 |  0.928571428571429 |  1600 |
| SCOTT  | 3000 |     20 |  0.214285714285714 |  3000 |
| FORD   | 3000 |     20 |  0.214285714285714 |  3000 |
| JONES  | 2975 |     20 |  0.285714285714286 |  3000 |
| ADAMS  | 1100 |     20 |  0.857142857142857 |  3000 |
| SMITH  |  800 |     20 |                  1 |  3000 |
+--------+------+--------+--------------------+-------+
14 rows in set (Elapsed: 00:00:00.17)
```

5.1.5.8.3 Other analysis functions

5.1.5.8.3.1 MDIFF function

grammar

```
MDIFF(colname , n , sortlist)
```

Table -597 Parameter Description

Parameter Name	explain
colname	Column names, constants, or expressions must be of integer type and cannot be null, '', or ''.
n	A positive integer constant with a value range of [14096].
sortlist	It can be a combination list of constants, column names, and multiple constants or columns, such as col1 desc, col2 asc. For the usage constraints of sortlist, please refer to the usage constraints of the order by clause in GBase 8a for the over clause. If you want to sort records, you can use the ASC or DESC keywords to specify the sorting rule, where ASC represents the ascending rule and DESC represents the

Parameter Name	explain
	descending rule. Sort records in ascending order by default

Function Description

The moving differential MDIFF function can sort the sorted result set according to the specified sorting rules, and calculate the difference between the current row and the previous n rows in the specified column.

When calculating, if there are not enough n rows in front, return a null value.

If the field of the specified column is numerical, the return value of the function is consistent with the field type and format of the specified column. If the calculation result exceeds the range of the current type, the processing rules are consistent with the system function lag() in 8a.

 **be careful**

Cannot appear in the search criteria of the WHERE clause.

Cannot be a parameter to an aggregation function.

Cannot be used in HAVING clause.

Cannot be used in GROUP BY LIST.

Cannot be nested inside other OLAP move functions and other non group by functions.

Cannot be used in non query parts of DELETE and UPDATE statements.

Example

Example: Based on the predetermined number of rows, use window movement on columns that support data types in the allowed usage positions.

```
SELECT MDIFF(y,3,x desc) from t2;

+-------------------+
| MDIFF(y,3,x desc) |
+-------------------+
|              NULL |
|              NULL |
|              NULL |
|                 2 |
|                 1 |
|                -2 |
```

```
|                          -3 |
|                           0 |
|                           0 |
|                           4 |
+--------------------+
10 rows in set (Elapsed: 00:00:00.04)


SELECT MDIFF(y,3,x) from t2;

+--------------+
| MDIFF(y,3,x) |
+--------------+
|         NULL |
|         NULL |
|         NULL |
|           -2 |
|           -2 |
|            0 |
|            3 |
|            2 |
|           -1 |
|           -2 |
+--------------+
10 rows in set (Elapsed: 00:00:00.04)
```

5.1.5.8.3.2 PIVOT function

grammar

```
PIVOT (<aggregate function>(<column to aggregate>), FOR col IN (constant))

PIVOT (<aggregate function>(<column to aggregate>), FOR (col, col,...) IN ((constant, constant,...),...))
```

For ease of explanation, PIVOT can be written as a concise syntax consisting of three parts of parameters:

```
PIVOT(
        pivot_ clause
        pivot_for_ clause
        pivot_ in_ clause
        )
```

Table 597 Parameter Description

Parameter Name	explain
pivot_ clause	Define the columns to aggregate, such as sum (score), avg (score), etc.
pivot_ for_ clause	Define the columns to rotate, such as FOR class.
pivot_ in_ clause	pivot_ for_ Column value of clause, pivot_ in_ The aggregation of each value in the clause will be converted into a separate column, such as IN ('math ','phy').

Function Description

PIVOT row to column function, used to rotate the column value of a certain column to the column name, that is, row to column conversion. This operation, also known as ' pivot ', involves rotating rows of data into columns in a crosstab.

be careful

No other functions can appear where the aggregate function is applied, or an error will be reported.

In pivot, there can be multiple aggregate function and aliases can be set for them. When there are multiple aggregate function, only one aggregate function is allowed to have no alias, and the alias cannot be duplicate, otherwise an error will be reported.

There can be multiple constants in IN in the pivot and aliases can be set.

In pivot, the alias set by the aggregate function will be merged with the alias set in IN and displayed as a new column name. If no alias is set for the aggregate function, the alias set in IN will be used. If no alias is set in IN, the value will be used as the alias by default. The naming rule of the new column is: "Alias of constant in IN _ Alias of aggregate function", so that the source of the new column name can be seen visually.

In the pivot, the constant in IN should be the column value after FOR.

Projection columns that do not appear in the pivot will serve as grouping conditions, which may result in the transformation being useless.

Example

```
select * from t3;
+-------+---------+------------+
| name | score    | class       |
+-------+---------+------------+
| xml    |      100 | math       |
| rjj     |    1100 | math       |
| rjj     |    1100 | chinese    |
| xml    |    1300 | chinese    |
| xml    |    1300 | pe           |
| rjj     |    1100 | pe           |
+-------+--------+------------+
6 rows in set (Elapsed: 00:00:00.03)

select * from t3 PIVOT(sum(score) for class in('math','pe'));

+-------+-------+-------+
| name | math | pe      |
+-------+-------+-------+
| xml    |    100 | 1300 |
| rjj     | 1100 | 1100 |
+-------+-------+-------+
2 rows in set (Elapsed: 00:00:00.42)

select * from t3 PIVOT(sum(score) for class in('math'as MATH,'pe'));

+-------+----------+-------+
| name | MATH | pe      |
+-------+---------+-------+
| xml    |      100 | 1300    |
| rjj     |    1100 | 1100    |
+-------+---------+--------+
2 rows in set (Elapsed: 00:00:00.09)

select * from t3 PIVOT(sum(score) as S for class in('math'as MATH,'pe'));

+-------+-------------+------+
| name | MATH_ S | pe_ S |
+-------+-------------+------+
| xml    |            100 | 1300 |
| rjj     |          1100 | 1100 |
+-------+-------------+------+
2 rows in set (Elapsed: 00:00:00.11)
```

```
select * from t3 PIVOT(sum(score),sum(score)as S for class in('math'as MATH,'pe'));

+-------+---------+--------+------------+-------+
| name | MATH | pe      | MATH_ S | pe_ S |
+-------+---------+--------+------------+-------+
| rjj     |    1100 |    1100 |        1100 |    1100 |
| xml    |      100 |    1300 |        100 |      1300 |
+-------+---------+--------+-----------+---------+
2 rows in set (Elapsed: 00:00:00.10)
```

5.1.5.8.3.3 UNPIVOT function

grammar

```
UNPIVOT[{INCLUDE|EXCLUDE}NULLS]
({COLUMN|(COLUMN|[,COLUMN]...)}
        FOR{COLUMN|(COLUMN|[,COLUMN]...)}
        IN ({COLUMN|(COLUMN|[,COLUMN]...)}
                [AS{LITERAL|(LITERAL[,LITERAL]...)}]
        [,{COLUMN|(COLUMN|[,COLUMN]...)}
                [AS{LITERAL|(LITERAL[,LITERAL]...)}]...]
))
```

For ease of explanation, UNPIVOT can be written as a concise syntax consisting of three parts of parameters:

```
UNPIVOT[{INCLUDE|EXCLUDE}NULLS]
(
        unpivot_ clause
        unpivot_for_ clause
        unpivot_ in_ clause
        )
```

Table 597 Parameter Description

Parameter Name	explain
unpivot_ clause	A new column name that stores the column values of the converted column.
unpivot_ for_ clause	A new column name that stores the column name of the converted column.
unpivot_ in_ clause	The columns to be converted have compatible column values.

Parameter Name	explain
INCLUDE NULLS	Display one row even if the data is NULL.
EXCLUDE NULLS	If the data is NULL, it will not be displayed. If not specified, it defaults to this parameter.

Function Description

The UNPIVOT column to row function is used to reverse pivot the data, that is, column to row. It can perform reverse pivot on the specified column, keeping the column values of other columns unchanged. At the same time, two new columns will be added, which store the column values and column names of the specified column, respectively.

be careful

unpivot_ in_ The columns in the clause must all belong to the same data type.

Only for unpivot_ in_ Set aliases for columns in clause.

Appear in unpivot_ in_ The columns in clause cannot appear for use in p rojection.

unpivot_ in_ The clause should be a column in the table.

Example

```
create table test(name varchar(20),math int, pe int);
Query OK, 0 rows affected (Elapsed: 00:00:00.09)

insert into test values('xml',100,92),('rjj',119,98);
Query OK, 2 rows affected (Elapsed: 00:00:00.28)
Records: 2  Duplicates: 0  Warnings: 0

select * from test;
+------+------+------+
| name | math | pe   |
+------+------+------+
| xml  |  100 |   92 |
| rjj  |  119 |   98 |
+------+------+------+
2 rows in set (Elapsed: 00:00:00.03)
```

```
select * from test unpivot(score for class in (math,pe));
+------+-------+-------+
| name | class | score |
+------+-------+-------+
| xml  | math  |   100 |
| rjj  | math  |   119 |
| xml  | pe    |    92 |
| rjj  | pe    |    98 |
+------+-------+-------+
4 rows in set (Elapsed: 00:00:00.09)

select * from test unpivot(score for class in (math as '1',pe));
+------+-------+-------+
| name | class | score |
+------+-------+-------+
| xml  | 1     |   100 |
| rjj  | 1     |   119 |
| xml  | pe    |    92 |
| rjj  | pe    |    98 |
+------+-------+-------+
4 rows in set (Elapsed: 00:00:00.03)

insert into test values('zs',null,null);
Query OK, 1 row affected (Elapsed: 00:00:00.27)

select * from test;

+------+------+------+
| name | math | pe   |
+------+------+------+
| xml  |  100 |   92 |
| rjj  |  119 |   98 |
| zs   | NULL | NULL |
+------+------+------+
3 rows in set (Elapsed: 00:00:00.03)

(The parameter include nulls is not supported)
```

5.1.5.9 ROWID function

Function Description

ROWID is a row number function that returns the row number information of a certain row of data on the data node.

Function Description

The ROWID function is similar to the primary key and is automatically maintained by the server without actual storage.

instructions

Implemented two grammars: pseudo column form and function form (both writing methods are completely equivalent and case insensitive). Among them, the function form is ROWID (table name).

Example:

```
SELECT *, ROWID, ROWID(t1) FROM t1;
SELECT * FROM t1 WHERE ROWID = 1;
```

Function Description

- ROWID return type is BIGINT, with numbering starting from 0;
- ROWID is equivalent to a pseudo column automatically added by the server to a table, which can be queried and used. It is automatically maintained by the server and does not require or allow users to manage it (such as modifying or creating indexes);
- ROWID is used as a reserved word, and it is not allowed to use ROWID as the name of any data object (regardless of case and whether there is a single prime);
- DML will not affect the ROWID of the original data;
- In terms of performance, a simple comparison between ROWID and constants, such as Where ROWID=1, intelligent indexes can be used to filter dcs. Supported comparison types include:>,>=,<,<=,=,<>, IS NULL, IS NOT NULL, BETWEEN, NOT BETWEEN;
- Only the Express engine supports ROWID, and other engines will report errors when executing (such as containing ROWID).

Example

Example 1: ROWID, ROWID (t1).

```
gbase> DROP TABLE IF EXISTS t1;
Query OK, 0 rows affected

gbase> CREATE TABLE t1(i int, j int);
Query OK, 0 rows affected

gbase> INSERT INTO t1 VALUES(2,1),(2,3),(2,3),(2,5),(3,2),(3,2),(3,2),(3,
```

```
4),(3,1),(3,5);
Query OK, 10 rows affected
Records: 10  Duplicates: 0  Warnings: 0

gbase> SELECT *, ROWID, ROWID(t1) FROM t1;
+------+------+-------+-------+
| i    | j    | ROWID | rowid |
+------+------+-------+-------+
|    2 |    1 |     0 |     0 |
|    2 |    3 |     1 |     1 |
|    2 |    3 |     2 |     2 |
|    2 |    5 |     3 |     3 |
|    3 |    2 |     4 |     4 |
|    3 |    2 |     5 |     5 |
|    3 |    2 |     6 |     6 |
|    3 |    4 |     7 |     7 |
|    3 |    1 |     8 |     8 |
|    3 |    5 |     9 |     9 |
+------+------+-------+-------+
10 rows in set
```

Example 2: ROWID=1

```
gbase> DROP TABLE IF EXISTS t1;
Query OK, 0 rows affected

gbase> CREATE TABLE t1(i int, j int);
Query OK, 0 rows affected

gbase> INSERT INTO t1 VALUES(2,1),(2,3),(2,3),(2,5),(3,2),(3,2),(3,2),(3,
4),(3,1),(3,5);
Query OK, 10 rows affected
Records: 10  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t1 WHERE ROWID = 1;
+------+------+
| i    | j    |
+------+------+
|    2 |    3 |
+------+------+
1  row in set
```

5.1.5.10 SEGMENT_ ID function

Function Description

SEGMENT_ ID(tbname)

The parameter tbname is the table name to obtain the node sharding number.

Function Description

Pseudo column, automatically maintained by the server and not actually stored. By parameter gcluster_ segment_ id_ Replace control, default value is 0, segment is not supported_ The id (tbname) function supports Segments when the value is set to 1_ The id (tbname) function.

instructions

When Sql is issued from gcluster to gnode, segment_ The id will be replaced with the sharding number.

Example:

```
set gcluster_ segment_ id_ replace=1;
select segment_ id(t) from t;
```

The statement received by N1 node is: select '1' as segment_ id(t1) from t1_ n1;

Function Description

- It can be used for the cluster layer to count the number of rows or other information for each shard;

  gbase> Select count(*) , segment_ id(t) from t group by segment_ id(t);

- Can be used in the cluster layer to query individual shards

  gbase> select * from t where segment_ id(t)='2';

- According to the sharding ID and ROWID, record uniqueness can be obtained.

  gbase> select * from t where segment_ id(t)='2' and rowid=1;

5.1.6 Full text search

5.1.6.1 Introduction to Full Text Retrieval

5.1.6.1.1 Product Introduction

summary

Full text search technology is to use all text sequences in various articles or information as search objects to find articles or objects containing the vocabulary to be searched for.

The GBase 8a MPP Cluster database supports full-text retrieval, with a default single word indexing method, supporting almost all languages, and ensuring a 100% query recall rate. Combining the unique column storage, compression, and intelligent indexing technology of GBase 8a MPP Cluster, it is suitable for retrieval and query applications targeting massive data.

5.1.6.1.2 major function

1. Establishing Indexing and Searching

- Supports indexing and querying of all text type fields in the table.
- Support parameterized management, and the process of index establishment, word segmentation, index maintenance, search, etc. can be conveniently configured through the standard configuration file of GBase 8a MPP Cluster.
- Embed text segmentation function in GBase 8a MPP Cluster to achieve single word segmentation of text columns and search strings, and ensure consistency in segmentation rules and results, preventing segmentation inconsistencies caused by contextual context.
- Supports full-text index synchronization queries, and supports query functionality during the index creation process. Newly added data can be indexed in batches. After the data in the index data buffer is processed and updated to the index file, users can immediately search for new content in these created indexes, instead of waiting for all new data to be indexed before querying.
- Supports logical expression queries (AND, OR, NOT) and NEAR queries for established full-text indexed columns in database tables, as well as logical combination queries with non full-text indexed fields.

2. Support DML

- Supports the deletion of character data type columns in database tables that have been full-text indexed.
- After updating the column data, the full-text index needs to be manually updated.

3. Supports DDL.

- Support the automatic invalidation of the database table index after the full-text index column is deleted.
- Supports database table renaming without invalidating the index.

> **explain**
>
> - For column renaming and data deletion, the processing method of GBase 8a MPP Cluster will not affect full-text indexing;
> - Full text retrieval currently supports UTF-8 and GBK encoding.
> - Full text queries do not support sub queries, that is, creating and using full-text indexes on sub queries is not supported.
> - The above functions provide standard SQL syntax support.

5.1.6.2 Full text retrieval example

summary

To use the full-text search function, it is necessary to first establish a table to store data source information, place the text content to be queried in the data table, and then create a full-text index for the text content column of the query. When there is content update in the table, the index should also be updated, so that the full-text search query syntax can be used for querying. The specific syntax includes operations such as creating an index, changing an index, updating an index, deleting an index, and querying. Below is an example of mobile SMS retrieval to quickly understand the usage of full-text retrieval. Please refer to the following chapters for specific syntax instructions.

Example

Example 1: Create a table SMS that stores SMS information, including two fields: phone number and SMS content.

```
gbase> CREATE TABLE sms (MB_No char(11), MB_ Text varchar(1000)
DEFAULT NULL);

Query OK, 0 rows affected
```

Example 2: Creating a full-text index with index name: idx_ t. Index column: MB_ Text。

```
gbase> CREATE FULLTEXT index idx_ t ON sms(MB_Text);
```

```
Query OK, 0 rows affected

Records: 0  Duplicates: 0  Warnings: 0
```

Insert sample data information

```
Insert INTO sms VALUES ('13023315123 ',' Nanjing University offers J ava programming training classes from 7:00 to 9:00 every Friday evenin g ');

Insert INTO sms VALUES ('13521000123 ', Tianjin University offers a postgraduate entrance examination training course with a semester of 3 months and classes every Saturday and Sunday. Welcome to register an d consult 40088800);

Insert INTO sms value ('13023315123 ',' Going to Nanjing University fo r English class in the evening, in Room 115 of the main building of Na njing University ');

Insert INTO sms Values ('13023315123 ',' Visit Tianjin Museum on wee kends and gather at the bus stop on Nanmen Outer Street in Nankai D istrict. ');

Insert INTO sms VALUES ('13023300023 ',' Our company is responsible for handling various documents, certificates/documents, absolute authent icity, phone number: 022-30088200. ');

Insert INTO sms Values ('13023300023 ',' Tomorrow, go to the company 's office cabinet to search for contract documents, files, and reimbursem ent vouchers. ');

Insert INTO sms Values ('13988213328 ',' Saturday morning at 9am, go to the water park, gather at the east gate, facing the Tianjin Tower ');


Insert INTO sms VALUES ('13323315181 ',' Nanjing University General Data Technology Co., Ltd. Address: Building J, Haitai Green Industry Base, No. 6 Haitai Development Sixth Road, Huayuan Industrial Zone, Tianjin (300384) ');

Insert INTO sms Values ('13521015341 ',' This primary school offers Ch inese, math, and English training courses for children aged 6 to 12 ');
```

Example 3: Manually updating the index

```
gbase> UPDATE INDEX idx_ t ON sms;

Query OK, 9 rows affected
```

Example 4: Index established by query (including full-text index)

```
gbase> SHOW INDEX FROM sms;

Due to the large number of columns in the result set, it is displayed in multiple
rows

+-------+------------+----------+--------------+-------------+

| Table | Non_ unique | Key_ name | Seq_ in_ index | Column_ name |

+-------+------------+----------+--------------+-------------+

| sms     |                 1 | idx_ t       |                    1 | MB_ Text         |

+-------+------------+----------+--------------+-------------+



+-----------+-------------+----------+--------+

| Collation | Cardinality | Sub_ part | Packed |

+-----------+-------------+----------+--------+

| NULL          |              NULL |         NULL | NULL     |

+-----------+-------------+----------+--------+



+------+------------+---------+

| Null | Index_ type | Comment |

+------+------------+---------+

| YES   |   FULLTEXT   |               |

+------+------------+---------+

1 rows in set
```

Example 5: Search Query

```
Gbase>SELECT COUNT (*) From sms WHERE containers (MB_Text,
```

```
'Nanjing University General Motors');
+----------+
| COUNT(*) |
+----------+
|        1 |
+----------+
1 row in set
```

Example 6: Query the number of data items that include text from Tianjin or training courses, but do not include text from "water":

```
Gbase>SELECT COUNT (*) From sms WHERE containers (MB_Text,
'Tianjin' | 'Training Course' - 'Water');
+----------+
| COUNT(*) |
+----------+
|        5 |
+----------+
1 row in set
```

Example 7: I would like to specifically check the text message content received by a mobile phone number starting with 135, which includes the phrase "Tianjin" or "training class" and does not include the phrase "water":

```
Gbase>SELECT Left (MB_Text, 30) From sms WHERE containers
(MB_Text, '("Tianjin" | "Training Class") - "Water") AND MB_ No like
'135%';
+------------------------------------------------------------+
| LEFT(MB_Text,30)                                           |
+------------------------------------------------------------+
|This primary school offers Chinese, mathematics, and English training courses for
children aged 6 to 12|
|Tianjin University offers postgraduate entrance examination training courses, with
```

```
a 3-month semester and classes every Saturday and Sunday. Welcome to register|
+------------------------------------------------------------+
2 rows in set
```

Example 8: Searching for SMS content that starts with the phrase 'Nanda':

```
Gbase>SELECT * From sms WHERE containers (MB_Text, '^ "Nanjing University");
+-------------+-----------------------------------------------------------------------------------------------------------+
| MB_ No              | MB_ Text
|
+-------------+-----------------------------------------------------------------------------------------------------------+
|13023315123 | Nanjing University offers Java programming training classes, which are held every Friday evening from 7:00 to 9:00|
|13323315181 | Nanda General Data Technology Co., Ltd. Address: Building J, Haitai Green Industry Base, No. 6 Haitai Development Sixth Road, Huayuan Industrial Zone, Tianjin (300384)|
+-------------+-----------------------------------------------------------------------------------------------------------+
2 rows in set
```

Example 9: To query the content of spam messages for creating fake diplomas:

```
gbase> SELECT MB_ Text from sms WHERE containers (MB_Text, 'Diploma'/2 ');
+------------------------------------------------------------+
| MB_ Text                                                   |
+------------------------------------------------------------+
|Tomorrow, go to the company's office cabinet to search for contract documents, files, and reimbursement vouchers|
|Our company handles various documents, certificates, and documents with
```

```
absolute authenticity. Contact number: 022-30088200|
+------------------------------------------------------------+
2 rows in set
```

Example 10: From the above information, it was found that normal SMS messages were also found. Modify the query statement to:

```
gbase> SELECT MB_ Text from sms WHERE containers (MB_Text, 'NEAR
(diploma, diploma), 4,1)');
+------------------------------------------------------------+
| MB_ Text                                                   |
+------------------------------------------------------------+
|Our company handles various documents, certificates, and documents with
absolute authenticity. Contact number: 022-30088200|
+------------------------------------------------------------+
1 row in set
```

Example 11: Query text messages containing the phrase 'Nanda' and sort them in reverse order of 'no'.

```
gbase> SELECT MB_ No AS no,MB_ Text from sms WHERE containers
(MB_Text, 'Nanjing University', 1) ORDER BY no DESC;
+-------------+------------------------------------------------------------+
| no                  | MB_ Text
|
+-------------+------------------------------------------------------------+
|13323315181 | Nanda General Data Technology Co., Ltd. Address: Building J,
Haitai Green Industry Base, No. 6 Haitai Development Sixth Road, Huayuan
Industrial Zone, Tianjin (300384)|
|13023315123 | Going to Nanjing University for English class in the evening, in
Room 115 of the main building of Nanjing University|
|13023315123 | Nanjing University offers Java programming training classes,
which are held every Friday evening from 7:00 to 9:00|
```

```
+-------------+-------------------------------------------------------------------+

3 rows in set
```

5.1.6.3 Full text search syntax

summary

On the basis of standard SQL statements, a syntax supporting full-text retrieval has been added, which is divided into two chapters. This chapter introduces the syntax for creating, modifying, manually updating, and deleting full-text indexes. The next chapter introduces the syntax used for querying full-text retrieval.

5.1.6.3.1 Create full-text search syntax

The establishment of GBase 8a MPP Cluster full-text index supports three modes:

- Specify full-text index columns when creating a table with the CREATE TABLE statement;
- CREATE FULLTEXT INDEX statement to establish a full-text index;
- ALTER TABLE... The ADD FULLTEXT INDEX statement establishes a full-text index.

5.1.6.3.1.1 Specify full-text index columns when creating a table with the CREATE TABLE statement

Grammar format

When users create tables, they can also create full-text indexes, which need to be created using the FULLTEXT keyword.

```
CREATE TABLE table_ name (

column defination… ,

fulltext [index] index_ name (column_name)

[INDEX_DATA_PATH='path']);
```

Table 534 Parameter Description

Parameter Name	explain
table_ name	Table name.
index_ name	Index name (the index name is table level unique and will not be case sensitive).
column_ name	Index column names, supporting CHAR, VARCHAR, or TEXT types.
INDEX_ DATA_ PATH	Optionally, set the index data path flag. If not filled in, the index data will be saved in the default path $GBASE_ BASE/userdata/gbase/dbname/metadata/tbname_ Under n *. GED: *. FTD, *. fti
path	Index data storage path, which should be an actual existing path.

be careful

The executing user needs to have read and write permissions to the directory specified by the path for storing index data.

Example

Example 1: When the index data storage path is not specified, it is stored in the default path.

```
gbase> DROP TABLE IF EXISTS sms;
Query OK, 0 rows affected
gbase> CREATE TABLE sms (MB_No char(11),MB_ Text varchar(1000)
DEFAULT NULL, FULLTEXT idx_ t (MB_Text));
Query OK, 0 rows affected
```

Example 2: Specify a storage path for indexed data. The steps are as follows (using gbase users to log in to the database as an example).

Step 1: The gbase user has read and write permissions to the directory where the index data is stored. You can use the root user to execute the following command to empower the gbase user:

```
# chown gbase:gbase /home
```

Step 2: Create a folder on each node to store index data.

```
# su - gbase
```

```
$ mkdir -p /home/fti
```

Step 3: Store the index data in the/home/fti/path.

```
gbase> CREATE TABLE text1 (col1 varchar(100), FULLTEXT INDEX fti_
col1 (col1) INDEX_ DATA_ PATH='/home/fti/');
Query OK, 0 rows affected
```

Example 3: The specified index data path does not exist, and the system reports an error message.

```
gbase> CREATE TABLE text1 (col1 varchar(100), FULLTEXT INDEX fti_
col1 (col1) INDEX_ DATA_ PATH='/index/dat/');
ERROR 1733 (HY000): (GBA-01EX-700) Gbase general error: Empty or invalid
index path
```

5.1.6.3.1.2 CREATE FULLTEXT INDEX statement to establish full-text indexing

Grammar format

After the user creates the table, use the FULLTEXT keyword to indicate which column in the table needs to be full-text indexed.

```
CREATE FULLTEXT INDEX index_ name ON table_ name (column_name)
[INDEX_DATA_PATH='path']
```

Table 535 Parameter Description

Parameter Name	explain
index_ name	Index name (the index name is table level unique and will not be case sensitive).
table_ name	Table name.
column_ name	Index column names, supporting CHAR, VARCHAR, or TEXT types.
INDEX_ DATA_ PATH	Optionally, set the index data path flag. If not filled in, the index data will be saved on the default path.
path	Index data storage path, which should be an actual existing path.

be careful

The executing user needs to have read and write permissions to the directory specified by the path for storing index data.

Example

Example 1: CREATE FULLTEXT INDEX

```
gbase> CREATE FULLTEXT INDEX idx_ t ON sms(MB_Text) INDEX_
DATA_ PATH='/home/fti/';
Query OK, 0 rows affected
Records: 0  Duplicates: 0  Warnings: 0
```

5.1.6.3.1.3 ALTER TABLE... ADD FULLTEXT INDEX statement to establish full-text indexing

Grammar format

After creating a table, indicate which column needs to be full-text indexed by modifying the table definition.

```
ALTER TABLE table_ name ADD FULLTEXT [INDEX] index_
name(column_name) [INDEX_DATA_PATH='path']
```

Table 536 Parameter Description

Parameter Name	explain
index_ name	Index name (the index name is table level unique and will not be case sensitive).
table_ name	Table name.
column_ name	Index column names, supporting CHAR, VARCHAR, or TEXT types.
INDEX_ DATA_ PATH	Optionally, set the index data path flag. If not filled in, the index data will be saved on the default path.
path	Index data storage path, which should be an actual existing path.

be careful

The executing user needs to have read and write permissions to the directory specified by the path for storing index data.

Example

Example 1: Modifying Table Definition in MB_ The Text column establishes a full-text index.

```
gbase>    DROP TABLE IF EXISTS sms;
Query OK, 0 rows affected

gbase> CREATE TABLE sms (MB_No char(11),MB_ Text varchar(1000) DEFAULT
NULL);
Query OK, 0 rows affected

gbase> ALTER TABLE sms add fulltext index idx_ t (MB_Text) INDEX_ DATA_
PATH='/home/fti/';
Query OK, 0 rows affected
Records: 0    Duplicates: 0    Warnings: 0
```

5.1.6.3.2 Delete full-text index

Grammar Description

After deleting the full-text index, the corresponding full-text index content will also be deleted and cannot be queried using the full-text index anymore.

GBase 8a MPP Cluster provides two syntax for deleting full-text indexes:

- DROP INDEX syntax delete full-text index
- Alter Table... DROP INDEX syntax delete full-text index

5.1.6.3.2.1 DROP INDEX syntax delete full-text index

Grammar format

```
DROP INDEX index_ name ON table_ name;
```

Table 537 Parameter Description

Parameter Name	explain
index_ name	Index name (the index name is table level unique, and the index name is not case sensitive).
table_ name	Table name.

Example

Example 1: Deleting a full-text index.

```
gbase> DROP INDEX idx_ t ON sms;

Query OK, 0 rows affected

Records: 0  Duplicates: 0  Warnings: 0
```

explain

After deleting the full-text index, the index content will also be deleted, so you cannot continue to use the full-text index function to query, otherwise the system will report an error.

```
Gbase>SELECT COUNT (*) From sms WHERE containers (MB_Text, 'Tianjin' | 'Training Course' - 'Water');

ERROR 1191 (HY000): Can't find FULLTEXT index matching the column list
```

5.1.6.3.2.2 ALTER TABLE... DROP INDEX syntax delete full-text index

Grammar format

```
ALTER TABLE table_ name DROP INDEX index_ name;
```

Table 538 Parameter Description

Parameter Name	explain
table_ name	Table name.
index_ name	Index name (including full-text index name).

Example

Example 1: Deleting a full-text index through an alter statement

```
gbase> ALTER TABLE sms DROP INDEX idx_ t;

Query OK, 0 rows affected

Records: 0  Duplicates: 0  Warnings: 0
```

5.1.6.3.3 Update full-text index

5.1.6.3.3.1 Overview of full-text index updates

summary

Full text index update refers to the synchronization operation between the data source content and the index content. In the event of changes in the data source content, the index also needs to be updated synchronously to maintain consistency. The method of updating the full text index is manual mode, which synchronizes the data and index content through corresponding SQL statements.

5.1.6.3.3.2 UPDATE INDEX syntax update full-text index

Grammar format

UPDATE INDEX *index_ name* ON *table_ name* [WITH ANALYZE];

Table -539 Parameter Description

Parameter Name	explain
index_ name	Index name (globally unique, case sensitive).
table_ name	Table name.
WITH ANALYZE	Analyzing instructions and updating full-text indexes after addition will reorganize discontinuous data, improve I/O speed, and thus improve performance.

Example

Example 1: Updating the SMS table named idx_ The full-text index of t.

```
gbase> UPDATE INDEX idx_ t ON sms;

Query OK, 0 rows affected
```

5.1.6.3.4 Query the existing full-text index

Grammar format

Query Table_ Index established on name (including full-text index).

```
SHOW INDEX FROM table_ name;
```

Table -540 Parameter Description

Parameter Name	explain
table_ name	Table name.

Example

```
gbase> SHOW INDEX FROM sms;
Due to the large number of columns in the result set, it is displayed in multiple
rows
+-------+------------+----------+--------------+-------------+
| Table | Non_ unique | Key_ name | Seq_ in_ index | Column_ name |
+-------+------------+----------+--------------+-------------+
| sms   |           1 | idx_ t    |            1 |      text     |
+-------+------------+----------+--------------+-------------+


+-----------+-------------+----------+--------+
| Collation | Cardinality | Sub_ part | Packed |
+-----------+-------------+----------+--------+
| NULL      |        NULL |     NULL | NULL   |
+-----------+-------------+----------+--------+


+------+------------+---------+
| Null | Index_ type | Comment |
+------+------------+---------+
| YES  |  FULLTEXT  |         |
+------+------------+---------+
1 rows in set
```

5.1.6.3.5 Full text retrieval supports unstructured data files

summary

Many scene applications of full-text retrieval are the search and query of big data files. These big data files have different formats, including HTML, doc, pdf, txt, XML, zip and other file formats, which belong to unstructured data files.

Since the field types of the database such as VARCHAR, BLOB and TEXT have upper limit constraints, they are not suitable for directly storing unstructured data files. Therefore, the URI type is added. Only the URI meta information of unstructured data files (including file storage path, file type, verification, etc.) is saved in the database, and the data file entities are stored in the file system outside the database, Realize the retrieval and query of unstructured data files through the content of URI.

The file formats supported by GBase 8a MPP Cluster for full-text retrieval URI types include HTML, doc, PDF, txt, XML, zip, and blob URI file formats. The content in these file formats can be parsed normally and full-text indexing can be established. For zip files, it is possible to parse the specific file content in the zip file compression package.

5.1.6.3.5.1 VARCHAR URI data type

The VARCHAR URI data type is implemented by adding a URI identifier to the VARCHAR type. The maximum length of the URI content is 2048 bytes, and the content has a certain data format.

For example:

```
CREATE TABLE fturi (fturi varchar (2048) URI);

INSERT INTO fturi VALUES (' file:///home/fti/dat/txt/189.txt
\r\nContent-Type:text/plain\r\n\r\n');
```

5.1.6.3.5.2 VARCHAR URI data format

VARCHAR URI data format: Its data is multi line text, separated by a pair of carriage returns and line feeds (" r n") between lines. The content is divided into three parts, and detailed explanations are shown in the table below:

Table 541 Data Format Content Description (mandatory options with *)

Content section	Parameter Name	explain

Content section	Parameter Name	explain
Part One*	URI	Protocol name "://" Authentication information directory file name ["?" Query parameters] ["#" Bookmark] The protocol name includes protocols such as file, HTTP, FTP, etc. The authentication information directory only supports absolute URI addresses and does not support relative addresses.
Part 2	Field List	Its basic form is "field name: field value", and the supported field names are described in the field list in Table **5-36**
Part Three*	Explicit end flag	A blank line used to indicate the end of the URI field data.

Table 5-42 Description of Field Names in the Field List

Field Name	explain
Content-Length	Used to indicate the size of data, formatted as a string of decimal digits. Content-Length = "Content-Length" ":" 1*DIGIT At least one number, for example: Content Length: 3495
Last-Modified	Indicate the date and time when the data was last modified. Last-Modified = "Last-Modified" ":" RFC 1123 HTTP-date For example: Last Modified: Tue, 15 Nov 1994 12:45:26 GMT
Content-MD5	MD5 verification. Content-MD5 = "Content-MD5" ":" md5-digest Md5 digest=<base64 encoding of 128-bit MD5 digest in RFC 1864>
Content-Type	Identify the media type. Content-Type = "Content-Type" ":" media-type media-type = type "/" subtype(";" " charset" "= " charset))

Field Name	explain
	type = token subtype = token charset = token For example: Content Type: text/html; charset=ISO-8859-4
Content-Encoding	Content Encoding is a modifier for the media type in Content Type. When present, its value indicates whether the data area is compressed, usually only appearing when the data area is plain text. Content-Encoding = "Content-Encoding" ":" "gzip" For example: Content Encoding: gzip

explain

- URI is the abbreviation of a Uniform Resource Identifier, used to uniquely identify a resource, including internet remote resources and local resources. The specific syntax specification is defined by RFC 3986.
- Content Length, Last Modified, and Content MD5 are all optional options. If they exist, the application and GBase 8a MPP Cluster should check whether the actual data size matches the description field when reading data. If they are found to be different, it indicates that the consistency of the data has been compromised. If they do not exist, consistency checks will not be performed.
- Content Type describes the data format. GBase 8a MPP Cluster will start the corresponding conversion plug-in according to the Content Type media type to correctly parse and convert the structured data obtained from the data file clock.

Example

```
INSERT INTO fturi VALUES ('
file:///tmp/fulltxt/subdirs_28323/e51029f0-e893-4836-a504-6d67804a6a0e
\r\nContent-Length:5571\r\nLast-Modified:Thu, 18 Oct 2012 11:21:21
GMT\r\nContent-MD5:ce0690d74cad8a310fc769b9ceb00153\r\nContent-Type:application/xml;
charset=utf8\r\n\r\n');
```

5.1.6.3.5.3 Summary of data formats supported by VARCHAR URI

1. Support file types

Table -543 File Types

file type	URI identification	explain
Txt	Content-Type:text/plain	
Pdf	Content-Type:application/pdf	
Word	Content-Type:application/msword	Only doc is supported, docx is not supported.
Zip	Content-Type:application/zip	Only the first file can be read, only the txt type is allowed.
Xml	Content-Type:application/xml	

2. Supports FTP files.

Examples are as follows (including password and no password methods):

```
INSERT INTO fturi VALUES (' ftp://ldy:liang999
@127.0.0.1/test/a01.txt\r\nContent-Type:text/plain\r\n
\r\n');
INSERT INTO fturi VALUES (' ftp://192.168.159.220/pub/a02.txt
\r\nContent-Type:text/plain\r\n\r\n');
```

3. Supports HTTP files.

For example (without password):

```
INSERT INTO fturi VALUES (' http://192.168.159.220/a03.txt
\r\nContent-Type:text/plain\r\n\r\n');
```

4. Escape character of URI file path

The URI path adopts a standard format and needs to be escaped when the path contains special characters.

Table -544 Standard Format

Serial number	Character (before escape, original)	Character after escape
one	" "	"%20"
two	"!"	"%21"
three	"\"	"%22"
four	"#"	"%23"
five	"%"	"%25"
six	":"	"%3A"
seven	";"	"%3B"
eight	"<"	"%3C"

nine	"="	"%3D"
ten	">"	"%3E"
eleven	"?"	"%3F"
twelve	"@"	"%40"
thirteen	"\\"	"%5C"
fourteen	"|"	"%7C"

5.1.6.3.5.4 Full text retrieval application

The URI data type is only an extension of the VARCHAR data type, so it is consistent with the VARCHAR data type in establishing, updating, and querying full-text indexes.

Example

Example: Application of full-text indexing.

Table creation statement:

```
CREATE TABLE fturi (fturi varchar (2048) URI);
```

Establish full-text index:

```
CREATE FULLTEXT INDEX idx_ 1 ON fturi(fturi);
```

Insert URI data

```
INSERT INTO fturi VALUES (' file:///home/fti/dat/txt/189.txt
\r\nContent-Type:
text/plain\r\n\r\n');
INSERT INTO fturi VALUES ('
file:///tmp/fulltxt/subdirs_28323/e51029f0-e893-4836-a504-6d67804a6a0e \r\
nContent-Length:5571\r\nLast-Modified:Thu, 18 Oct 2012 11:21:21
GMT\r\nContent-MD5:ce0690d74cad8a310fc769b9ceb00153\r\nContent-Type:
application/xml; charset=utf8\r\n\r\n');
……
```

Update full-text index:

```
UPDATE INDEX idx_ 1 ON fturi;
```

Query full-text index fields.

```
SELECT * From fturi WHERE containers (fturi, 'Sohu');
```

5.1.6.3.6 Full text retrieval supports word segmentation types

summary

The content of an index, which is a text string, is composed of a series of word sequences, including Chinese, English letters, and numbers.

There are two built-in word segmentation methods in the entire text: natural word segmentation and multivariate word segmentation.

For English letters, it is also possible to set whether they are case sensitive.

These are all set through configuration files, and the configuration file paths on the coordinator and data nodes are as follows:

$GCLUSTER_ HOME/lib/gbase/plugin/gbfti/cfg/GbaseCharExt.xml

$GBASE_ HOME/lib/gbase/plugin/gbfti/cfg/GbaseCharExt.xml

5.1.6.3.6.1 Set word segmentation type

Natural word segmentation is to divide words according to the type of text, and separate them naturally by spaces and punctuation mark.

The corresponding configuration item parameters are as follows: (0: natural participle; 1: numerical multiple participle; 2: English multiple participle; 3: numerical and English multiple participle;...):<multisegmask>0</multisegmask>

Example

Original text:

```
The R&D personnel have changed a total of 112314 lines of code, including 53 bugs in the restored and closed states.
```

Example 1: Chinese is split according to single characters.

```
Research/development/personnel/personnel/transformation/replacement/generation /code/line/quantity/total/112314/line/its/middle/restored/closed/status/BUG/has/53/ units
```

Example 2: Multivariate participle, such as ternary participle, which mainly targets English and numbers, treating three consecutive characters as a single term.

```
Research/development/personnel/personnel/transformation/replacement/generation /code/line/quantity/total/112/123/231/314/line/its/middle/res/so/sov/ovl/vle/led/clo/ los/ose/sed/status/BUG/yes/53/
```

5.1.6.3.6.2 Is case sensitive

Mainly targeting English letters.

The corresponding configuration item parameters are as follows (0: indistinguishable; 1: distinguished):<mixedcase>0</mixedcase>

5.1.6.4 Full text index query syntax

summary

Full text retrieval is mainly applied in fields such as text matching, internet search engines, and information retrieval. For example, searching for sensitive vocabulary in text messages, filtering spam messages, searching for keywords on the internet, intelligent sorting, and other application scenarios all involve querying text content. When querying, it is possible to query a single word or multiple words, and multiple words can also be combined using logical operation operators. Before querying, it is necessary to first understand the concepts of words, word order, and word spacing.

5.1.6.4.1 Words, word order, word spacing

1. Item:

   Also known as the minimum search unit. In Western language, words are separated by spaces, and the smallest unit of retrieval is a word. For example, 'cat and mouse' is three words. For multi byte encoded text, such as Chinese, GBase 8a MPP Cluster defaults to using a single word as a search unit because there is no clear separator between words to distinguish them. For example, "Shanghai" defaults to two words in full-text search, namely "Shang" and "Hai". When searching (Shang&Hai), both Shanghai and Hai will hit, and the two words are not guaranteed to be adjacent. If you want to be adjacent, you need to explicitly set "Shanghai". When analyzing text, spaces and line breaks between Western words are directly filtered out, and are not used as index words or calculated for occupied positions. The spaces between Chinese characters and other symbols (such as full width punctuation mark such as "",!; ￥ [], and special characters such as #, *, $, etc.) are also filtered and are not used as index words. However, the positions occupied by these symbols will be recorded when creating indexes, which will affect word spacing.

2. Word order:

Refers to the order between two words. When querying multiple words, it is necessary to specify whether the multiple words to be queried are ordered or unordered, and filter the relevant conditions by specifying the word order. For example, to search for "Shanghai" without word order requirements, not only "Shanghai" can be found, but also "sea" can be found.

3. Word spacing:

   Refers to the number of words separated between two words, and also includes the first and last words being queried. For example, in the text 'the black cat catch white mouse', to query for 'cat mouse', the word spacing is 4. For Chinese, the word spacing is the number of characters between two Chinese characters (characters include Chinese characters, punctuation, spaces, symbols, etc., and carriage returns are omitted). For example, if you want to search for "Shanghai" in the text "Last week, I went to a meeting at the Oceanic Bureau...", the word spacing is 6. For example, if you search for "diploma" in the spam message "selling and producing fake documents * # certificates, sending * tickets", the word spacing is 4.

4. The particularity of spaces:

   When calculating word spacing, the processing of spaces between Chinese and English is different. The spaces between English and Chinese are filtered out as separators and do not occupy a space. For example, when searching for "the great", the word spacing between these two words is 2, while the space between Chinese and Chinese takes up a space. For example, when searching for "Tianjin", the word spacing between these two words is 3, which is different from "Tianjin" without spaces.

5.1.6.4.2 CONTAINS() function

Use the SELECT statement to query text content, where the CONTAINS() function is used as the query condition to quickly match the column content with full-text indexing set.

Query syntax:

```
SELECT query_ column FROM table_ name

WHERE CONTAINS (column_name, Query Content[, score_flag])
```

Table 545 Parameter Description

Parameter Name	explain
query_ column	The query result columns that need to be displayed.
table_ name	Table name.
column_ name	Query columns, which are column names that have created full-text indexes, do not support multi column queries.
Query Content	The content to be searched is character type and requires the use of '...' Reference query content. The content supports text queries, logical operation query expressions, NEAR functions, and more. The maximum length supported for content is 255.
score_ flag	SCORE label, optional. If left blank, it means no evaluation value will be output and should be used in conjunction with the SCORE function.

 **explain**

When a query statement contains multiple contains function conditions, score_ The flag parameter values should be different from each other, or a syntax error message will be reported:

```
Incorrect arguments to CONTAINS FUNCTION - SCORE FLAG
COLLISION。
```

Example

Example 1: How many SMS messages contain the four characters "Nanjing University General".

```
Gbase>SELECT COUNT (*) From sms WHERE containers (MB_Text,
'Nanjing University General Motors');
+----------+
| COUNT(*) |
+----------+
|        1 |
+----------+
1 row in set

Gbase>SELECT COUNT (*) From sms WHERE containers (MB_Text,
```

```
'Nanjing University General Motors', 0);
+----------+
| COUNT(*) |
+----------+
|        1 |
+----------+
1 row in set
```

Example 2: Query text messages containing 'Nankai' and sort them in reverse order of 'no'.

```
gbase> SELECT MB_ No AS no,MB_ Text from sms WHERE containers
(MB_Text, 'Nanjing University', 1) ORDER BY no DESC;
+-------------+-----------------------------------------------------------------------------------
-------------------------------------------------+
| no                  | MB_ Text
|
+-------------+-----------------------------------------------------------------------------------
-------------------------------------------------+
|13323315181 | Nanda General Data Technology Co., Ltd. Address: Building J,
Haitai Green Industry Base, No. 6 Haitai Development Sixth Road, Huayuan
Industrial Zone, Tianjin (300384)|
|13023315123 | Going to Nanjing University for English class in the evening, in
Room 115 of the main building of Nanjing University|
|13023315123 | Nanjing University offers Java programming training classes,
which are held every Friday evening from 7:00 to 9:00|
+-------------+-----------------------------------------------------------------------------------
-------------------------------------------------+
3 rows in set
```

Example 3: The contains function is valid in where conditions and does not support searching in having conditions, meaning it is invalid in having conditions.

```
gbase> SELECT MB_ TEXT From sms WHERE containers (MB_Text,
'Nanjing University');
+-----------------------------------------------------------+
| MB_ Text                                                  |
+-----------------------------------------------------------+
|Nanjing University offers Java programming training classes every Friday evening
from 7:00 to 9:00|
|I will go to Nanjing University for English classes in the evening, in Room 115 of
the main building of Nanjing University|
|Visit Tianjin Museum over the weekend and gather at the bus stop on Nanmenwai
Street in Nankai District|
|Nanda General Data Technology Co., Ltd. Address:|
+-----------------------------------------------------------+
```

```
4 rows in set (Elapsed: 00:00:00.02)

gbase> SELECT MB_ TEXT FROM sms GROUP BY MB_ TEXT having
containers (MB_Text, 'Nanjing University');
ERROR 1149 (42000): contains function can ONLY in where clause and on clause
```

be careful

The Contains function can only be used in Where clauses. If it is included in other types of clauses, an error will be reported:

```
Gbase>SELECT containers (a, 'universal') From t WHERE containers
(a, 'China') AND score (0)>10;

ERROR 1149 (42000): contains function can ONLY in WHERE clause
```

5.1.6.4.3 Query expression syntax

5.1.6.4.3.1 All syntax of query expressions

1. Explicit AND operator '&', such as hello&world;
2. Implicit and (AND) operator 'space', such as' hello world ';
3. Or (OR) operator '|', such as hello | world;
4. Non (NOT) operator '-', such as hello world;
5. The prefix operator '^', such as ^ hello;
6. The suffix operator '$', such as' mouse $';
7. Phrase query operator '', such as' Nanda ';
8. Grouping operator (), such as (hello world)&(cat | dog);
9. The threshold matching character '/', such as' the great wall is a wonderful place '/3;
10. NEAR search function near (term1, term2), num, order), such as near ((great, place), 2, 1), where num represents word spacing, order 0 represents no word order, and order 1 represents word order;
11. Extension option: The search expression is divided into two parts: basic expression and extension option. The total length is limited to 255 characters, and the extension option can be empty. Currently, the extension option only supports rank=tf,

indicating that the correlation algorithm uses word frequency instead of the default BM25 algorithm. For example, "Nanda: rank=tf" indicates searching for Nanda, and the correlation is word frequency.

5.1.6.4.3.2 Priority of query operators

When querying, operators in the expression syntax can be combined, and the priority of the operators is as follows:

priority	Same priority within the same row
High "-" Low	/
	^ $
	"" near ()
	|
	-
	&

explain

- The query content defaults to AND operation, for example, 'aaa bbb' is equivalent to 'aaa&bbb';
- The OR operation level is higher than the AND operation, for example, 'aaa&bbb | ccc' is equivalent to 'aaa&(bbb | ccc)';
- Individual non query content has no meaning, and search engines do not execute corresponding queries, such as' - bbb ', and the engine does not query;
- The particularity of spaces is that in query statements, except for the&| - ^ $() operation character, consecutive characters are treated as a whole, and only spaces are used as separator identifiers. For example, "Tianjin Beijing" is equivalent to "(Tianjin) - (Beijing)". When creating a full-text index, spaces between words in English are used as separators, but placeholders are not calculated. For example, "the great" is the word "the" and "great", with a word spacing of 2; For spaces between Chinese characters, placeholders are calculated, such as "Tianjin" in Chinese, where the word spacing between the two characters in Tianjin is 3;
- At present, punctuation mark are not involved in indexing and searching. They only serve as placeholders when indexing. Punctuation mark is calculated when calculating word spacing.

5.1.6.4.3.3 Explicit AND operator '&'

Operator Meaning

The queried content must be fully included.

Example

For example, 'hello&world' indicates that the query contains both hello and world words.

5.1.6.4.3.4 Implicit AND operator 'space'

Operator Meaning

The queried content must be fully included.

Example

For example, 'hello world' indicates that the query contains both hello and world words.

5.1.6.4.3.5 OR operator '|'

Operator Meaning

At least one of the queried content is included.

Example

For example, 'hello | world' indicates querying for content containing hello or world words.

5.1.6.4.3.6 Non (NOT) operator '-'

Operator Meaning

The queried content must not be included.

Example

For example, 'hello world' represents querying for content that contains hello but does not contain world words.

explain

If only text in the form of "- log" is queried without any other query criteria, the query cannot be executed because it includes almost all indexed documents.

5.1.6.4.3.7 Phrase query operator ''

Operator Meaning

The content in double quotation marks "" will be viewed as a phrase for querying.

Example

For example, when querying "Nanda", it means that the query content includes the phrase "Nanda" (which includes the words "Nanda" and "Da" and is adjacent to each other, in the order of "Nanda").

5.1.6.4.3.8 Initial operator '^'

Operator Meaning

To query information that starts with a certain word, you can add a ^ symbol before the query word, and note that there should be no spaces between the ^ symbol and the word. For example, when querying '^ south', it means querying information starting with a south character.

Example

Example: Searching for SMS content that starts with a phrase from Nanda University

```
Gbase>SELECT * From sms WHERE containers (MB_Text, '^ "Nanjing University");
+-----------+-----------------------------------------------+
|no                | text                                                                  |
+-----------+-----------------------------------------------+
|13023315123 | Nanjing University offers Java programming training classes, which are held every Friday evening from 7:00 to 9:00|
|13323315181 | Nanjing University General Data Technology Co., Ltd. Address:|
+-----------+-----------------------------------------------+
```

```
2 row in set
```

5.1.6.4.3.9 Final word operator '$'

Operator Meaning

To search for information that ends with a certain word, you can add a $symbol after the query word, and note that there should be no spaces between the $symbol and the word. For example, querying 'class $' means querying information that ends with a class character.

Example

Example: Searching for SMS content that ends with a training class phrase

```
Gbase>SELECT * From sms WHERE containers (MB_Text, 'Training Class' $');
+-------------+-------------------------------------------------+
| MB_ No              | MB_ Text                                                                  |
+-------------+-------------------------------------------------+
|13521015341 | Our primary school offers Chinese, mathematics, and English training classes for children aged 6 to 12|
+-------------+-------------------------------------------------+
1 row in set
```

5.1.6.4.3.10 Grouping operator ()

Operator Meaning

Group multiple query criteria into queries, and then concatenate the query results using logical operators (OR, AND, NOT).

For example, (hello world)&(cat | dog) can group multiple query conditions and then concatenate the query results using logical operators (OR, AND, NOT).

Example

Example 1: Query the number of SMS messages that include text from Tianjin or training courses but do not include text from "water".

```
Gbase>SELECT COUNT (*) From sms WHERE containers (MB_Text,
'Tianjin' | 'Training Course' - 'Water');
+----------+
| COUNT(*) |
+----------+
|                5 |
+----------+
1 row in set
```

Example 2: Query SMS content that includes both Nanjing University and training classes.

```
Gbase>SELECT * From sms WHERE containers (MB_Text, 'Nanda
Training Class');
+-----------+-------------------------------------------------+
|no                | Text                                                                  |
+-----------+-------------------------------------------------+
|13521015341 | Nanjing University offers Java programming training classes every
Friday evening from 7:00 to 9:00|
+-----------+-------------------------------------------------+
1 row in set
```

Example 3: Query text messages containing text from Tianjin or Nanjing University, including training courses.

```
gbase> SELECT MB_ Text from sms WHERE containers (MB_Text,
'Tianjin' | 'Nanjing University'&'Training Class');
+------------------------------------------------------------+
| text                                                       |
+------------------------------------------------------------+
|Tianjin University offers postgraduate entrance examination training courses, with
a 3-month semester and classes every Saturday and Sunday. Welcome to register|
+------------------------------------------------------------+
1 rows in set
```

The result is the same as the following query statement.

```
gbase> SELECT MB_ Text from sms WHERE containers (MB_Text, '("
Tianjin" | "Nanjing University")&"Training Class");
+------------------------------------------------------------+
| text                                                       |
+------------------------------------------------------------+
|Nanjing University offers Java programming training classes every Friday ev
ening from 7:00 to 9:00|
|Tianjin University offers postgraduate entrance examination training courses,
with a 3-month semester and classes every Saturday and Sunday. Welcome t
o register|
+------------------------------------------------------------+
2 rows in set
```

5.1.6.4.3.11 Threshold matching character '/'

Operator Meaning

When querying multiple keywords, the number of keywords that meet the matching criteria. English words are separated by spaces, while Chinese words are separated by one word. The query content needs to be enclosed in quotation marks.

For example, if the query statement condition is "the great wall is a wonderful place"/3, it means that as long as the words in three of the queries are met, the condition is met.

Example

```
gbase> SELECT MB_ Text FROM sms WHERE contains(MB_Text, '"th
e great wall is a wonderful place"/3');
+-------------------------------------------------------------+
| MB_Text                                                     |
+-------------------------------------------------------------+
| the great wall is so wonderful ,i like this place.          |
+-------------------------------------------------------------+
1 row in set
```

5.1.6.4.3.12 NEAR search function

Grammar format

```
NEAR ((term1, term2), num[,  Order])
```

Table 546 Parameter Description

Parameter Name	explain
term	For search terms, regardless of whether they are marked with quotation marks, search according to phrases, such as near ((Beijing, Tianjin), 10) equivalent to near ("Beijing", "Tianjin"), 10), term can also be a Near expression, with two terms separated by commas. English is the word, while Chinese defaults to a single word.
NUM	Indicates the number of word spacings, non zero integers, word spacings (including matching words), and actual word spacings less than num are considered symbol requirements
Order	Indicates word order. A value of 0 represents no word order, and a value of 1 represents with word order. Order is optional, with a default value of 0, indicating no word order.

For example, near (great, place), 3,1), search for words with a word spacing of no more than 3 between great and place, and search in word order.

1. Text matching condition description

- Enter the value according to the parameter order:
  - 0: Include all query words in no order;

- Non zero integer: includes all query words in order.

- The distance between matching words should not exceed the expected value (including matching words);
- Supports recursive matching, and each layer of recursive results must meet the conditions (1) and (2). The recursive results are passed as a whole to the next recursive calculation, and their length is the matching length.

2. Search instance description

- Enter the query statement 'near ((cat, dog), 5, 1)':
  - The text 'cat dog' - matches correctly in word order;
  - Text 'dog cat' - Mismatch, dog cat word order mismatch;
  - The text "cat aaa bbb ccc ddd dog" - does not match, there are 6 words (including matching words) between cat dogs, exceeding the matching length by 5.
- Enter the query statement 'near ((cat, dog), 5, 0)':
  - Text "cat dog" - matching;
  - The text 'dog cat' - matches without word order requirements.
  - Text 'dog aaa bbb ccc ddd cat' - Mismatch, there are 6 words (including matching words) between dog and cat, exceeding the matching length by 5
- Enter the query statement 'near ((near (cat, dog), 5,1), mouse), 8,0)'
  - Text "cat dog mouse" - matching;
  - Text "mouse cat dog" - matching;
  - Text 'dog cat mouse' - Mismatch, dog, cat word order mismatch;
  - The text "cat aaa bbb ccc ddd dog mouse" - does not match, cat, dog word spacing is 6, exceeding the matching length of 5;
  - Text 'cat aaa bbb ccc dog aaa bbb ccc ddd mouse' - Mismatch, cat, mouse word spacing of 10 exceeds the matching length of 8;
  - The text "cat aaa bbb ccc dog ddd mouse" - matches, with cat and dog word spacing not exceeding, and cat and mouse word spacing not exceeding;
  - The text 'mouse ddd cat aaa bbb ccc dog' - matches, with cat and dog word spacing not exceeding, and mouse and dog word spacing not exceeding;

The above are examples of syntactic explanations, and the following are actual examples to illustrate.

Example

Example 1: To query the content of spam messages that create fake diplomas.

```
gbase> SELECT MB_ Text from sms WHERE containers (MB_Text, 'Di
ploma'/2 ');
+-------------------------------------------------------------+
| MB_Text                                                     |
+-------------------------------------------------------------+
|Tomorrow, go to the company's office cabinet to search for contract docum
ents, files, and reimbursement vouchers|
|Our company handles various documents, certificates, and documents with a
bsolute authenticity. Contact number: 022-30088200|
+-------------------------------------------------------------+
2 rows in set
```

We have also found normal SMS messages from the above information. Let's improve the query method.

```
gbase> SELECT MB_ Text from sms WHERE containers (MB_Text, 'N
EAR (diploma, diploma), 4,1)');
+-------------------------------------------------------------+
| MB_Text                                                     |
+-------------------------------------------------------------+
|Our company handles various documents, certificates, and documents with a
bsolute authenticity. Contact number: 022-30088200|
+-------------------------------------------------------------+
1 row in set
```

Example 2: Querying SMS containing two characters from Nanda with a word spacing of 4.

```
gbase> SELECT MB_ Text from sms WHERE containers (MB_Text, '(N
EAR (South, Great), 4,0)');
+-------------------------------------------------------------+
| MB_ Text
          |
+-------------------------------------------------------------+
|I will go to Nanjing University for English classes in the evening, in Roo
m 115 of the main building of Nanjing University|
|Visit Tianjin Museum over the weekend and gather at the bus stop on Nan
menwai Street in Nankai District|
|Nanda General Data Technology Co., Ltd. Address:|
```

```
|Nanjing University offers Java programming training classes every Friday ev
ening from 7:00 to 9:00|
+------------------------------------------------------------------+
4 rows in set
```

Example 3: I want to query a text message containing two characters from Nanda, with a word spacing of 4 and ending with a class character.

```
gbase> SELECT MB_ Text from sms WHERE containers (MB_Text, '(NEAR
(South, University), 4,0))&lesson $');
+-----------------------------------------------------+
| MB_ Text                                            |
+-----------------------------------------------------+
|Nanjing University offers Java programming training classes every Friday evening
from 7:00 to 9:00|
+-----------------------------------------------------+
1 row in set
```

5.1.6.4.4 SCORE evaluation function

Grammar format

This function calculates the score based on the matching degree of keywords in full-text retrieval. The returned value is the weight value of the full-text query function CONTAINS query result, and the size of the value is related to the full-text retrieval evaluation algorithm. The higher the weight value, the higher the matching degree. The default evaluation mode for GBase 8a MPP Cluster full-text retrieval is the BM25 algorithm, which can be set to use the word frequency evaluation algorithm through the extension option "rank=tf".

```
INT SCORE(N)
```

Table 547 Parameter Description

Parameter Name	explain
INT	The return value is of type INT, which is the weight of the full-text query function CONTAINS query result consistent with its SCORE label,
N	SCORE label, integer type, which should be consistent with the SCORE label of a full-text query CONTAINS function in the Where clause of the current query statement.

be careful

- The SCORE function can only be used in projection columns, GROUP BY, and ORDER BY clauses, otherwise an error will be reported.

Gbase>SELECT * From t WHERE containers (a, 'China', 0) AND score (0)>10;

ERROR 1149 (42000): score function can ONLY in SELECT, GROUP BY or ORDER BY clause

- If there is no full-text query function CONTAINS that matches its SCORE label, an error message will be reported:

Gbase>SELECT score (1) a1 From t WHERE containers (a, 'China', 0);

ERROR 1149 (42000): not match the number of score func

- If the SCORE labels of multiple CONTAINS functions are consistent with them, an error message is reported:

gbase> SELECT score(0) FROM t WHERE contains(b, 'abc' , 0) AND contains(b, 'bcb' , 0);

ERROR 1210 (HY000): Incorrect arguments to CONTAINS FUNCTION - SCORE FLAG COLLISION

Example

Example 1: Query text messages containing the phrase "Nanda" and sort them in reverse according to the score. In the example, the weight value in the first row of data is higher compared to the other two rows, because the query for "Nanda" in the first row hit twice, while the other two rows only hit once.

Gbase>Insert INTO sms VALUES (1,1, 'Going to Nanjing University for English classes in the evening, in Room 115 of the main building of N anjing University');

Query OK, 1 row affected

Insert INTO sms Values (1,1, 'Address of Nanda General Data Technolo

```
gy Co., Ltd.: Building J, Haitai, No. 6, Haitai Development, Huayuan In
dustrial Zone, Tianjin');

Query OK, 1 row affected

Gbase>Insert INTO sms Values (1,1, 'Nanjing University offers Java pro
gramming training classes from 7:00 to 9:00 every Friday evening');

Query OK, 1 row affected

gbase> SELECT score(1) AS score, MB_ Text from sms WHERE contai
ners (MB_Text, 'Nanjing University', 1) ORDER BY score DESC;

+-------+-----------------------------------------------------+

| score | MB_ Text
      |

+-------+-----------------------------------------------------+

      |1511 | In the evening, I will attend an English class at Nanjing Unive
rsity in Room 115 of the main building|

      |1508 | Nanjing University General Data Technology Co., Ltd. Address:
|

      |1508 | Nanjing University offers Java programming training classes, wi
th classes every Friday evening from 7:00 to 9:00|

+-------+-----------------------------------------------------+

3 rows in set
```

5.1.6.5 Full text indexing performance tuning

summary

Full text indexing is a universal component program that varies in data size and hardware configuration for different users. In order to fully leverage the effectiveness of full-text indexing, personalization was achieved by modifying configuration files.

The paths of this configuration file on the coordinator node and data node are as follows:

```
$GCLUSTER_ HOME/lib/gbase/plugin/gbfti/cfg/GbaseCharExt.xml

$GBASE_ HOME/lib/gbase/plugin/gbfti/cfg/GbaseCharExt.xml
```

5.1.6.5.1 Update Index Optimization

summary

The dictionary of full-text indexing is stored in a static hash structure, and updating the index requires multiple queries to the hash table. When the number of words is too large, the conflict chain of the dictionary is very long, and a long conflict chain can cause the update speed to slow down sharply.

5.1.6.5.1.1 Modify the number of threads

Updating the full-text index adopts a multi-threaded parallel approach, which is specifically divided into threads such as word segmentation, sorting, and output. When the server has a large number of CPU cores, the performance can be optimized by modifying these three parameter values. When users want to fully utilize CPU resources, it is recommended to set these three parameters to CPU cores * 3/4 to achieve optimal system performance. The corresponding configuration item parameters are as follows:

```
<segThreads>4</segThreads>

<sortThreads>4</sortThreads>

<outThreads>3</outThreads>
```

5.1.6.5.1.2 Modify the number of dictionary hash buckets

The dictionary for full-text indexing is stored in a static hash structure. When there are too many words, the conflict chain of the dictionary is very long, and updating the index requires multiple queries of the dictionary. A long conflict chain can cause the update speed to slow down sharply. The default hash bucket length is 65536 * 256. The corresponding configuration item parameters are as follows:

```
<dictSlotPerUnit>16777216</dictSlotPerUnit>
```

5.1.6.5.1.3 The Meaning of Quick Update Flag

Changing the number of hash buckets in a full-text index can effectively improve the speed of updating the full-text index. However, when there are too many words in the dictionary and the document content is too large (especially in the URI mode), it still cannot meet the needs of users, and this mode can be adopted. In this mode, the index data is written to multiple files simultaneously, which can effectively solve the IO waiting bottleneck and greatly improve the speed. The corresponding configuration item parameters are as follows (0: Do not update quickly; 1: Update quickly):

```
<quickUpdate>0</quickUpdate>
```

5.1.6.5.2 Index Query Optimization

summary

Full text indexing is database based, and single database queries are executed serially (i.e. single threaded). In order to fully utilize the multi-core resources of the system and accelerate query speed, it can be divided into multiple libraries, and the parallelism (i.e. the maximum number of execution threads) can be reasonably set to achieve the goal of controlling system resources.

5.1.6.5.2.1 Modify the number of single libraries

The number of a single full-text database is defined as one unit, which is a complete and independent full-text index structure that includes dictionaries and all inverted information. Scoring and sorting are also calculated within this library, independent of other libraries. Single database queries are conducted in a serial manner. The corresponding configuration item parameters are as follows:

```
<maxDocPerUnit>100000000</maxDocPerUnit>
```

5.1.6.5.2.2 Modify query parallelism

Query parallelism is the maximum number of execution threads allowed for the current query, with each parallel group defined as a task. The corresponding configuration item parameters are as follows:

```
<maxThreadPerTask>5</maxThreadPerTask>
```

For example, improving query performance through database partitioning: If the current table data is 200 million and the server has 8 cores, it is recommended to set maxThreadPerTask to 8 and maxDocPerUnit to 25000000 to maximize server query performance.

5.1.6.5.2.3 Improving query performance by executing analysis full-text commands

Full text index data is stored in blocks, which can cause discontinuous inverted data after multiple updates to the index, leading to increased read write jumps and thus affecting query performance. Adding analysis instructions when updating the index will promptly reorganize discontinuous data, improve IO speed, and thus improve performance. The syntax format of this command is as follows:

```
UPDATE INDEX index_ name ON table_ name WITH ANALYZE;
```

5.1.6.5.2.4 Improving query performance by making full-text libraries resident in memory

Full text index data occupies a considerable amount of memory, especially when the number of words is large. Reading full text index data from disk into memory is a relatively time-consuming operation. When the user's server memory is large, it is

recommended to use memory resident mode (default mode), which saves loading time and improves performance. The corresponding configuration item parameters are as follows (0: memory resident; 1: non memory resident):

```
<reduceMemMode>0</reduceMemMode>
```

5.1.6.6 Table operation affects indexing

When performing DDL and DML operations on the table where the full-text index is located, the full-text index will also make corresponding changes.

Table 548 Comparison Table of the Impact of Table Operations on Index

Operation on the table where the index is located	The impact of full-text indexing
Change the name of the table where the index is located	The index will change the index data directory name based on the new table name
Index column deleted	Index Follow Delete
Column data update	Update when updating the index
Column data deletion	Indexing is not processed and relies on GBase 8a MPP Cluster's original deletion mechanism to mask query results

5.1.7 Transaction Control

5.1.7.1 Function Description

The GBase 8a MPP Cluster standard transaction currently mainly implements the following functions:

1. Implement a unified standard transaction mechanism to ensure the ACID characteristics of transactions;

Atomicity: the modification of data by transactions must be atomic, either all or none.

Consistency, when a transaction is completed, must ensure that all data remains in a consistent state.

Isolation: Modifications made by concurrent transactions must be isolated from any other concurrent transactions. The ANSI/ISOSQL92 standard defines the isolation levels for some database operations: Read uncommitted, Read committed, Repeatable read, and Serializable. The transaction isolation level supported by GBase 8a MPP Cluster is the snapshot level.

Durability: The impact on the database system after a transaction is completed is

persistent, and data can be restored even if the database goes down.

2. Support transaction standard syntax:

START TRANSACTION | BEGIN

COMMIT

ROLLBACK

3. Supports the following DML operations: Insert Values, UPDATE, DELETE, Insert... SELECT, MERGE.
4. Support continuous writing within transactions;
5. Supports concurrent queries and DML on the same table, but does not support concurrent write operations on the same table (insert values and insert values can be concurrent);
6. Support the inclusion of hash indexes and full-text indexes in tables;
7. Support table storage quota mechanism;
8. Support table row and column attributes;
9. Supports load data loading operations.

5.1.7.2 Parameter configuration

To enable the transaction function in GBase 8a MPP Cluster, configuration parameters need to be added to the following files for all cluster nodes

- At GNode node $GBASE_ BASE/config/gbase_ 8a_ Add the following parameters to the gbase.cn configuration file

```
gbase_ tx_ log_ mode=USE,STANDARD_ TRANS
```

- At GCluster node $GCLUSTER_ BASE/config/gbase_ 8a_ Add the following parameters to the gcluster.cnf configuration file

```
gcluster_ transaction_ disable=0
```

be careful

After modifying the configuration file, it is necessary to restart the cluster for the transaction function to take effect.

5.1.7.3 Transaction Standard Syntax

Grammar format

```
[START TRANSACTION | BEGIN] [COMMIT] [ROLLBACK]
```

Table 549 Parameter Description

Parameter Name	explain
START TRANSACTION | BEGIN	Start a new transaction.
COMMIT	Submit the current transaction to make the change permanent.
ROLLBACK	Roll back the current transaction and cancel its changes.

Example

```
gbase> create table t1(a int);
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> start transaction;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> insert into t1 values(10);
Query OK, 1 row affected (Elapsed: 00:00:00.00)
gbase> commit;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> SELECT * from t1;
+------+
| a    |
+------+
|   10 |
+------+
1 row in set (Elapsed: 00:00:00.00)
gbase> begin;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> insert into t1 values(200);
Query OK, 1 row affected (Elapsed: 00:00:00.01)
gbase> rollback;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
gbase> SELECT * from t1;
+------+
```

```
| a       |
+------+
|     10 |
+------+
1 row in set (Elapsed: 00:00:00.01)
```

5.1.8 DDL syntax

Data Definition Language, a language used to define and manage all objects in an SQL database. The DDL language of GBase 8a Cluster MPP includes CREATE, ALT, DROP, and TRUNCATE operations on objects such as DATABASE, TABLE, VIEW, INDEX, etc.

Table -550 Object and Supported DDL Operation Instructions

object	Object Meaning	Supported DDL operations
DATABSE	database	CREATE、DROP
TABLE	surface	CREATE、ALTER、DROP、TRUNCATE
VIEW	view	CREATE、ALTER、DROP
INDEX	Indexes	CREATE、ALTER、DROP

Table -551 DDL Operation Instructions

DDL operation	DDL operation function
CREATE	create object
ALTER	Modifying Object Properties
DROP	delete object
TRUNCATE	Delete records within an object

5.1.8.1 DATABASE

5.1.8.1.1 CREATE DATABASE

Function Description

CREATE DATABASE is the creation of a database with the given name.

Grammar format

```
CREATE DATABASE [IF NOT EXISTS] [vc_name.]database_ name;
```

Table -552 Parameter Description

Parameter Name	explain
vc_ name	The VC name of the database to be created.
database_ name	The name of the database to be created.

> **explain**
>
> Users need to obtain permission to create a database in order to use CREATE DATABASE.

Example

```
gbase> CREATE DATABASE IF NOT EXISTS mydb;
Query OK, 1 row affected
```

5.1.8.1.2 DROP DATABASE

Function Description

DROP DATABASE deletes the specified database and the tables it contains.

Grammar format

```
DROP DATABASE [IF EXISTS] [VC_NAME.]database_ name;
```

Table -553 Parameter Description

Parameter Name	explain
IF EXISTS	Prevent reporting errors when deleting databases due to non-existent databases
vc_ name	The VC name of the database to be created.
database_ name	The name of the database to be created.


warning

Please use this statement with caution! DROP DATABASE will delete all tables in the specified database. Users need to strengthen the management of database DROP permissions and have DROP permissions for all tables, views, and other objects in the specified database in order to successfully execute DROP DATABASE operations.

Example

Example 1: Using keywords to delete a database.

```
gbase> DROP DATABASE IF EXISTS test;
Query OK, 1 row affected
```

5.1.8.2 TABLE

5.1.8.2.1 CREATE TABLE

Function Description

Create a table in the current database with the name given by the user. Users must have permission to create tables.

Grammar format

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS]
[vc_name.][database_name.]table_ name
(column_definition [ , column_definition],    ... [,    key_options])
[table_options]
[partition_options]

column_ definition:
column_ name data_ type [NOT NULL | NULL] [DEFAULT default_value]
[COMMENT 'comment_value']
key_ options:
KEY "index_name" ("column_name") KEY_ BLOCK_ SIZE=<VALUSE> KEY_
DC_ SIXE=<dc_ value> USIONG HASH GLOBAL
table_ options:
[COMMENT 'comment_value']

partition_ options:
```

```
    PARTITION BY
        { RANGE(expr)
          | LIST(expr)
          | LINEAR] HASH(expr)
          | [LINEAR] KEY(column_list)}
    [PARTITIONS num]
    [SUBPARTITION BY
        { [LINEAR] HASH(expr)
        | [LINEAR] KEY(column_list) }
      [SUBPARTITIONS num]
    ]
    [(partition_definition [, partition_definition] ...)]
partition_ definition:
    PARTITION partition_ name
        [VALUES
            {LESS THAN {(expr | value_list) | MAXVALUE}
            |
            IN (value_list)}] [COMMENT 'comment_value']
        [(subpartition_definition [, subpartition_definition] ...)]
subpartition_ definition:
    SUBPARTITION logical_ name
```

Table 554 Parameter Description

Parameter Name	explain
TEMPORARY	This parameter is optional and is required to create a temporary table.
PARTITION BY	This parameter is optional. This keyword is used when creating a partition table. Please refer to the section "CREATE TABLE PARTITION" for creating a partition table.
IF NOT EXISTS	This parameter is optional and users can use the keyword IF NOT EXISTS to create a table. If the table already exists, the system will report WARNING information.
vc_ name	This parameter is optional. After specifying a virtual cluster, create databases and tables under this virtual cluster. If the specified VC is not displayed_ The name parameter indicates that the created database belongs to USE VC VC_ Virtual cluster after name.
database_ name	This parameter is optional. After specifying the database, create a table under it. If the specified database is not displayed_ Name

Parameter Name	explain
	parameter, the created table belongs to the USE database_ The table in the database after name.
table_ name	Please refer to the section on "Database, Table, Index, Column, and Alias" for table naming rules. By default, tables are created in the current database. If the current database is not specified or the table already exists, an error message is reported.
column_ name	Specify the data columns in the table.
data_ type	Specify the data type of the data column. Data types refer to the content in "Data Types".
NOT NULL | NULL	Specify whether NULL is allowed for the value of the data column. If neither NULL nor NOT NULL is specified, the column is considered to have specified NULL.
default_ value	Specify the default value for the data column. The default value must be a constant, not a function or an expression. For example, users cannot set the default value of a data column to NOW () or Current_ Functions such as DATE(). For a given table, the SHOW CREATE TABLE statement can be used to see which columns have an explicit DEFAULT clause.
column_ definition comment_ value	Specify a note description for the data column. For example: stu_ No id COMMENT 'Student ID'.
key_ options	Specify the hash index in the table.
index_ name	Specify the index name to create
index_ column_ name	Specify the name of the index column to create
KEY_ DC_ SIZE=dc_ value	Specify the segmentation size of the HASH index. dc_ The maximum value is 2147483646, and the minimum value is 0. The default is 0 to create a HASH index on the entire column.
USING HASH GLOBAL	Specify the HASH index as a global index.
table_ options	The default is a random distribution table.
REPLICATED	Create a replicated table using the keyword REPLICATED. The

Parameter Name	explain
	replicated table stores complete data on each data node of GBase 8a MPP Cluster.
DISTRIBUTED BY *column_ name*	Specify the physical column in the creation table_ The name is a hash column, and the table created in this way is called a hash distribution table. Hash columns can be of integer type (INT, BIGINT, etc.), VARCHAR, or DECIMAL type
table_ options: COMMENT	Specify a note description for the table. You can use SHOW CREATE TABLE table_ Name and SHOW FULL COLUMNS From table_ Use the name statement to display note information. The annotation length limit for a table is 2000 bytes.
COMPRESS	Specify table level data compression attributes, including compression attribute values for character types and compression attribute values for numerical types.

be careful

- The tail of a replicated table name is not allowed to be_ End of n {number}, such as mytable_ n1， mytable_ N12 is not allowed to be used.
- When performing OGG Kafka data synchronization, the value of the HASH distribution column must not be empty

Example

Example 1: Create a regular table.

```
gbase> CREATE TABLE t1(a int , b varchar(50) NOT NULL DEFAULT 'gbase');
Query OK, 0 rows affected

gbase> DESC t1;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | YES  |     | NULL    |       |
| b     | varchar(50) | NO   |     | gbase   |       |
+-------+-------------+------+-----+---------+-------+
```

```
2 rows in set
```

Example 2: Create a table with column annotation information.

```
gbase> DROP TABLE IF EXISTS t1;
Query OK, 0 rows affected

gbase> CREATE TABLE t1 (f_username varchar(10) comment 'name' );
Query OK, 0 rows affected

gbase> DESC t1;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| f_ username | varchar(10) | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
1 row in set

gbase> SHOW CREATE TABLE t1;
+-------+-----------------------------------------------------+
| Table | Create Table                                        |
+-------+-----------------------------------------------------+
| t1    | CREATE TABLE "t1" (
  "f_username" varchar(10) DEFAULT NULL COMMENT 'name'
)                                                             |
+-------+-----------------------------------------------------+
1 row in set
```

5.1.8.2.1.1 CREATE TABLE... AS SELECT...

Function Description

Create a table structure based on column definitions and projected columns, and copy the data queried in SELECT into the created table.

Grammar format

```
CREATE TABLE [vcname.][dbname.] table_ name [(column_definition,...)]
[REPLICATED | DISTRIBUTED BY (col_name) ] [AS] SELECT ...;
```

Table 555 Parameter Description

Parameter Name	explain
AS	Specify a SELECT statement with optional keywords.

Please refer to the parameter description section in the "CREATE TABLE" section for other parameter descriptions.

Example

Example 1: Copying the entire table structure and data to create a randomly distributed table.

```
gbase> CREATE TABLE t7(a int, b decimal(10,5), c float, d datetime);
Query OK, 0 rows affected

gbase> INSERT INTO t7 VALUES(1,2.234,3.345,'2019-11-11 11:11:11'),
(3,4.567,5.678,'2019-11-11 22:22:22');
Query OK, 2 rows affected
Records: 2  Duplicates: 0  Warnings: 0

gbase> CREATE TABLE t8 SELECT * FROM t7;
Query OK, 2 rows affected

gbase> SELECT * FROM t8;
+------+---------+-------+---------------------+
| a    | b       | c     | d                   |
+------+---------+-------+---------------------+
|    1 | 2.23400 | 3.345 | 2019-11-11 11:11:11 |
|    3 | 4.56700 | 5.678 | 2019-11-11 22:22:22 |
+------+---------+-------+---------------------+
2 rows in set
```

Example 2: Copy partial table structure and data by column to create a hash distribution table distributed by the specified column.

```
gbase> CREATE TABLE t9 distributed by('a') SELECT a,b FROM t7;
Query OK, 2 rows affected
Records: 2  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t9;
+------+------+
| a    | b    |
+------+------+
|    1 |    2 |
|    3 |    5 |
+------+------+
2 rows in set
```

Example 3: Create a randomly distributed table by filtering and copying partial table structures and data according to conditions.

```
gbase> DROP TABLE IF EXISTS t10;
```

```
Query OK, 0 rows affected

gbase> CREATE TABLE t10 SELECT a,b FROM t7 where d>'2019-11-11 11:11:11';
Query OK, 1 row affected
Records: 1  Duplicates: 0  Warnings: 0

gbase>  SELECT * FROM t10;
+------+------+
| a    | b    |
+------+------+
|    3 |    5 |
+------+------+
1 row in set
```

Example 4: Create a random distribution table by copying partial table structures and data based on the results of a joint query.

```
gbase> DROP TABLE IF EXISTS t11;
Query OK, 0 rows affected

gbase> CREATE TABLE t11 SELECT a,b FROM t7 where d>'2019-11-11 11:11:11' UNION ALL SELECT a,b FROM t7 where d='2019-11-11 11:11:11';
Query OK, 2 rows affected
Records: 2  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t11;
+------+------+
| a    | b    |
+------+------+
|    3 |    5 |
|    1 |    2 |
+------+------+
2 rows in set
```

5.1.8.2.1.2 CREATE TABLE... LIKE...

Function Description

Copy table_ Create a table using the table structure of name2_ name1。

Grammar format

```
CREATE TABLE [vc_name.][db_name.]table_ name1 LIKE table_ name2;
```

Example

Example 1: Create a random distribution table.

```
gbase> DROP TABLE IF EXISTS t5;
Query OK, 0 rows affected

gbase> CREATE TABLE t5(a int,b datetime);
Query OK, 0 rows affected

gbase> INSERT INTO t5 VALUES(1,NOW());
Query OK, 1 row affected

gbase> CREATE TABLE t6 LIKE t5;
Query OK, 0 rows affected

gbase> SHOW CREATE TABLE t6;
+-------+-------------------------------------------------------------+
|                Table                |                Create                Table
|
+-------+-------------------------------------------------------------+
| t6    | CREATE TABLE "t6" (
  "a" int(11) DEFAULT NULL,
  "b" datetime DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace' |
+-------+-------------------------------------------------------------+
1 row in set

gbase> SELECT * FROM t6;
Empty set
```

5.1.8.2.1.3 CREATE TEMPORARY TABLE ...

Function Description

When creating a table, users can use the keyword TEMPORARY. The temporary table is limited to the current connection, and will be automatically deleted when the connection is closed. This means that two different connections can use the same temporary table name without conflicts or conflicts with an existing table with the same name (the existing table will be hidden until the temporary table is deleted).

Grammar format

```
CREATE TEMPORARY TABLE [IF NOT EXISTS] [vc_name.][db_name.]table_
name [(column_definition,...)] [REPLICATED | DISTRIBUTED BY
```

```
(col_name) ] ;
```

be careful

- Temporary tables support all DDL and DML operations except for ALT.
- Temporary tables cannot be backed up.
- Temporary tables support exporting data from tables using query result export statements in the current connection.

Example

Example 1: Creating a temporary table.

```
gbase> CREATE TEMPORARY TABLE tem_ table (a int);
Query OK, 0 rows affected

gbase> INSERT INTO tem_ table (a) values (1);
Query OK, 1 row affected

gbase> INSERT INTO tem_ table (a) values (2);
Query OK, 1 row affected

gbase> SELECT * FROM tem_ table;
+------+
| a    |
+------+
|    1 |
|    2 |
+------+
2 rows in set

gbase> EXIT;
Bye
$ gbase -uroot -p
Enter password:

GBase client 9.5.3.17.117651. Copyright (c) 2004-2020, GBase.  All Rights Reserved.
gbase> USE test;
Query OK, 0 rows affected
```

```
gbase> SELECT * FROM tem_ table;
ERROR 1146 (42S02): Table 'test.tem_ table' doesn't exist
```

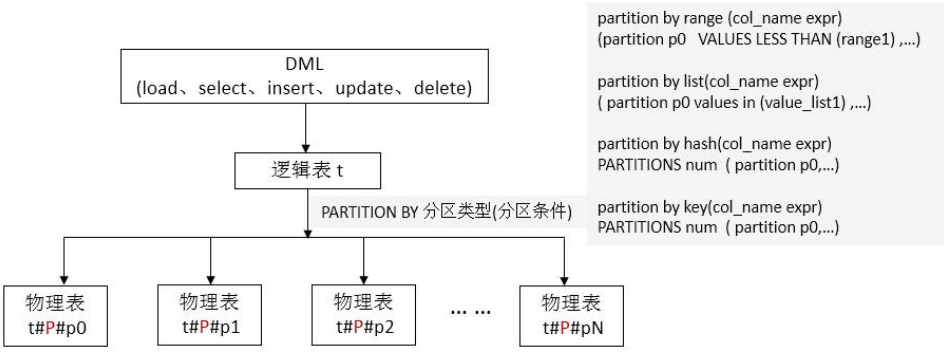
5.1.8.2.1.4 CREATE TABLE PARTITION

5.1.8.2.1.4.1 Overview of partition table

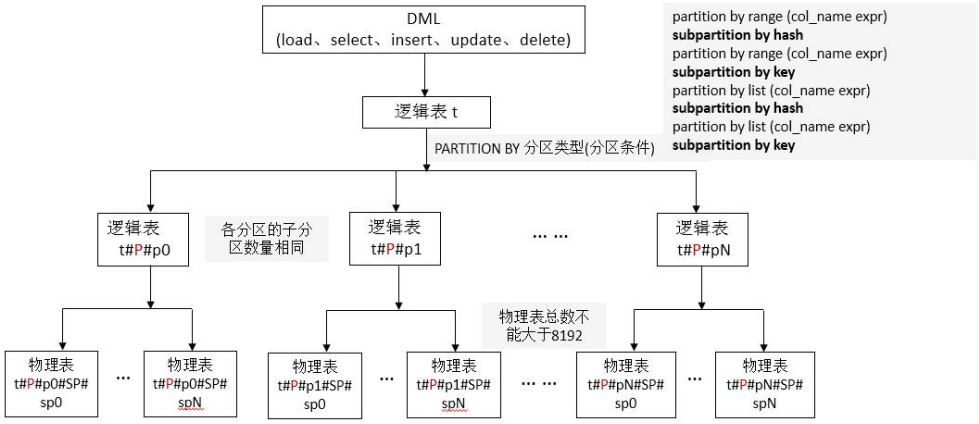
Function Description

Partition table decomposes a table in the database into multiple smaller parts that are easy to manage according to certain rules. Logically, there is only one table, but the bottom layer is composed of multiple physical partitions.

The currently supported partitions include RANGE partition, LIST partition, HASH partition, and KEY partition.



RANGE partition table and LIST partition table support sub partitions, which can only be HASH partitions or KEY partitions.



Partition management currently supports creating a partition table, deleting a partition table, adding partitions to a partition table, deleting partitions from a partition table, truncating a partition table to specify partition data, and renaming a partition table to specify partition names. The support of partition table functions varies from version to version, and each function has usage restrictions. For specific operation syntax and usage restrictions, refer to the detailed instructions in the corresponding chapters of the manual.

System table information_ The created partition table information can be queried in schema.partitions.

- RANGE partition

Range partitioning. Customize multiple ranges as partitions from small to large, with no overlapping areas between each partition (range). The table data that needs to be stored in the warehouse is calculated according to the specified partition conditions and stored in the corresponding range of partitions. Data that is not within all partition ranges cannot be saved and stored.

If the corresponding partition condition in the receipt data has a NULL value, RANGE partition table will insert the NULL value into the first partition.

RANGE partition supports HASH sub partition and KEY sub partition, that is, the data in each partition of RANGE partition table can be subdivided according to the HASH or KEY partition algorithm, and the data will be stored in the corresponding sub partition. The number of sub partitions in each partition of the RANGE partition table is the same.

The partition condition partition by range (colname expr) for RANGE partitions can be:

(1) A specified column that supports tinyint, int, smallint, bigint, date, datetime, time

(2) An expression based on the column specified in (1) and returning an integer. The allowed operators and functions in expressions are shown in the table below.

abs()	ceiling()	ceil()	datediff()	day()	dayofmonth()
dayofweek()	dayofyear()	floor()	hour()	microsecond()	minute()
year()	month()	quarter()	second()	time_to_sec()	to_days()
from_days()	weekday()	yearweek()	extract()	+	-
*	mod	div	%		

For example:

```
gbase> CREATE TABLE t1 (a datetime) PARTITION BY RANGE
(dayofmonth(a)) (PARTITION p0 VALUES LESS THAN (10));

Query OK, 0 rows affected

gbase> CREATE TABLE t1 (a int) PARTITION BY RANGE (a mod 10)
(PARTITION p0 VALUES LESS THAN (10));

Query OK, 0 rows affected
```

- LIST partition

List partition. Customize multiple value lists as partitions, with no overlapping values between each partition (value list). There is no need for duplicate values within each partition (value list). The table data that needs to be entered is calculated according to the specified partition conditions and stored in the corresponding partition of the value list. Data that is not in the list of all partition values cannot be saved and stored.

If the partition condition corresponding to the inbound data has a NULL value, the LIST partition must contain a NULL value, otherwise an error will be reported when inserting a NULL value.

LIST partition supports HASH sub partition and KEY sub partition, which can subdivide the data in each partition of LIST partition table according to the HASH or KEY partition algorithm, and store the data in the corresponding sub partition. The number of sub partitions in each partition of LIST partition table is the same.

The partition condition partition by LIST (colname expr) for a LIST partition can be:

(1) A specified column that supports tinyint, int, smallint, bigint, date, datetime, time

(2) An expression based on the column specified in (1) and returning an integer. The allowed operators and functions in expressions are shown in the table below.

abs()	ceiling()	ceil()	datediff()	day()	dayofmonth()
dayofweek()	dayofyear()	floor()	hour()	microsecond()	minute()
year()	month()	quarter()	second()	time_to_sec()	to_days()
from_days()	weekday()	yearweek()	extract()	+	-
*	mod	div	%		

For example:

```
gbase> CREATE TABLE t1 (a datetime) PARTITION BY list (time_to_sec(a))
(partition p0 values in (3,5,6,9,17));

Query OK, 0 rows affected

gbase> CREATE TABLE t1 (a int) PARTITION BY list (floor(a div 10))
(partition p0 values in (3,5,6,9,17));

Query OK, 0 rows affected
```

- HASH partition

HASH partition. Customize the number of partitions, calculate the table data that needs to be stored according to the specified partition conditions, and use the HASH algorithm to store the data as evenly as possible in each partition.

HASH partitions do not support sub partitions.

The partition condition by HASH (colname expr) for HASH partitions can be:

(1) A specified column that supports tinyint, int, smallint, bigint

(2) An expression based on the column specified in (1) and returning an integer. The allowed operators and functions in expressions are shown in the table below.

abs()	ceiling()	ceil()	datediff()	day()	dayofmonth()
dayofweek()	dayofyear()	floor()	hour()	microsecond()	minute()
year()	month()	quarter()	second()	time_to_sec()	to_days()
weekday()	yearweek()	extract()	+	-	*
mod	div	%			

For example:

```
gbase>  CREATE TABLE t1 (a int) PARTITION BY hash(abs(a));
Query OK, 0 rows affected
gbase>  CREATE TABLE t1 (a int) PARTITION BY hash(day(a));
Query OK, 0 rows affected
gbase>  CREATE TABLE t1 (a int) PARTITION BY hash(a%10);
Query OK, 0 rows affected
```

- KEY partition

KEY partition. Customize the number of partitions, calculate the table data that needs to be stored according to the specified partition conditions, and use the internal fixed HASH algorithm to store the data as evenly as possible in each partition.

KEY partition does not support sub partitions.

The KEY partition uses the internal fixed HASH algorithm. The KEY partition table is different from the HASH partition table:

- KEY supports multiple column lists in partition conditions, while HASH only supports one column
- KEY does not support functions and expressions in partition conditions, HASH supports functions and expressions for a column
- There are more types of columns supported by KEY in partition conditions

The partition condition partition by KEY (colname expr) for a KEY partition can be:

(1) A list of a specified column or multiple specified columns

(2) (1) The type of column referred to in supports tinyint, int, smallint, bigint, float, double, decimal, date, datetime, time, timestamp, char, varchar

For example:

```
gbase> create table t1 (a varchar(100)) partition by key(a);
Query OK, 0 rows affected
gbase> create table t1 (a decimal) partition by key(a) partitions 10;
Query OK, 0 rows affected
gbase> create table t1 (a int,b varchar(100),c datetime) partition by key(a,b,c)
```

```
partitions 10;

Query OK, 0 rows affected
```

be careful

- The total number of partitions, including sub partitions, shall not exceed 8192;
- When creating the HASH partition table and KEY partition table, the number of partitions is not specified. A partition is created by default.
- In the process of creating the partition table, the disk space is insufficient and an error is reported;
- In the process of creating a partition table, the partition name is duplicate and an error is reported;
- In the process of creating a partition table, the partition name does not conform to the naming convention, and an error is reported. The naming convention of the partition table is consistent with that of the common table;
- It is not supported to specify different tablespaces for each partition;
- The partition name is not specified when creating the partition table. The default partition name is p0, p1;
- Partition column does not support update operation;
- The NULL value is placed in the first partition of the RANGE partition table; The LIST partition table is placed in the partition whose list value contains NULL. If all list values do not contain NULL, the NULL insertion error is reported; The partitions inserted into the NULL of the HASH and KEY partition table are uncertain.
- The number of sub partitions in each partition must be the same;
- Only range and list partitions can create sub partitions.

5.1.8.2.1.4.2Create RANGE partition table

Grammar format

```
partition_ options:

        PARTITION BY RANGE(expr)

        (partition_definition [, partition_definition] ...)

      [SUBPARTITION BY]

            {   [LINEAR] HASH(expr)
```

```
    |   [LINEAR] KEY(column_list) } [COMMENT 'comment_value']
    [SUBPARTITIONS num]
partition_ definition:
      PARTITION partition_ name
            VALUES LESS THAN {(expr) | MAXVALUE}
          [(subpartition_definition [, subpartition_definition] ...)]
subpartition_ definition:
    SUBPARTITION logical_ name
```

explain

- Expr is a column value or an expression based on a column value that returns an integer value;
- The value of the expr list for each partition must be incremented.
- The types of columns and the functions and operators supported by expressions refer to the specific descriptions in the overview.
- The length limit for table annotations is 2000 bytes. In information_ When querying in schema.parts, only 80 bytes of annotation information are displayed, and the rest are truncated and not displayed. Show create table can display the complete annotation content.

Example

Example 1: Create a RANGE partition table.

```
gbase> CREATE TABLE t1 (
            a int(11) DEFAULT NULL,
            b varchar(10) DEFAULT NULL
        ) REPLICATED PARTITION BY RANGE (a)
(Partition p0 VALUES LESS THAN (10) comment 'Partition one',
Partition p1 VALUES LESS THAN (20) comment 'Partition two',
Partition p2 VALUES LESS THAN (30) comment 'Partition three',
Partition p3 VALUES LESS THAN (40)) comment 'Partition four'
Query OK, 0 rows affected (Elapsed: 00:00:00.11)
```

Annotations can be viewed through the following statement:

select length(PARTITION_COMMENT) from information_ schema.partitions where PARTITION_ NAME = 'p0';

Example 2: Create a RANGE partition table with hash sub partitions and no sub partition name specified.

```
gbase> create table t1 (id int, dt int)
        partition by range (id)
        subpartition by hash (quarter(dt))
        subpartitions 4
        (
        partition p0 values less than (1990),
        partition p1 values less than (2000),
        partition p2 values less than maxvalue
        );
Query OK, 0 rows affected (Elapsed: 00:00:00.17)
```

Example 3: Create a RANGE partition table with hash sub partitions and specify the sub partition name.

```
gbase> create table t1 (id int, dt int)
        partition by range (id)
        subpartition by hash (quarter(dt))
        (
        partition p0 values less than (1990)
        (
        subpartition part0_ In the first quarter,
        subpartition part0_ q2,
        subpartition part0_ q3,
        subpartition part0_ q4
```

```
    ),
    partition p1 values less than (2000)
    (
    subpartition part1_ q1,
    subpartition part1_ In the second quarter,
    subpartition part1_ q3,
    subpartition part1_ q4
    ),
    partition p2 values less than maxvalue
    (
    subpartition part2_ q1,
    subpartition part2_ q2,
    subpartition part2_ q3,
    subpartition part2_ Four Seasons
    )
    );
Query OK, 0 rows affected (Elapsed: 00:00:00.16)
```

Example 4: Create a RANGE partition table with key sub partitions and no sub partition name specified.

```
gbase> create table t1 (id int, dt date)
    partition by range (id)
    subpartition by key (dt)
    subpartitions 4
    (
    partition p0 values less than (1990),
    partition p1 values less than (2000),
    partition p2 values less than maxvalue
```

```
        );
Query OK, 0 rows affected (Elapsed: 00:00:00.16)
```

Example 5: Create a RANGE partition table with a key sub partition and specify the sub partition name.

```
gbase> create table t1 (id int, dt date)
        partition by range (id)
        subpartition by key (dt)
        (
        partition p0 values less than (1990)
        (
        subpartition part0_ In the first quarter,
        subpartition part0_ q2,
        subpartition part0_ q3,
        subpartition part0_ q4
        ),
        partition p1 values less than (2000)
        (
        subpartition part1_ q1,
        subpartition part1_ In the second quarter,
        subpartition part1_ q3,
        subpartition part1_ q4
        ),
        partition p2 values less than maxvalue
        (
        subpartition part2_ q1,
        subpartition part2_ q2,
        subpartition part2_ q3,
```

```
        subpartition part2_ Four Seasons

        )

        );

Query OK, 0 rows affected (Elapsed: 00:00:00.16)
```

5.1.8.2.1.4.3Create LIST partition

Grammar format

```
partition_ options:

    PARTITION BY LIST(expr)

    (partition_definition [, partition_definition] ...)

[SUBPARTITION BY]

  {   [LINEAR] HASH(expr) |   [LINEAR] KEY(column_list) }

[SUBPARTITIONS num]

partition_ definition:

    PARTITION partition_ name

        VALUES IN (value_list)

    [(subpartition_definition [, subpartition_definition] ...)]

subpartition_ definition:

    SUBPARTITION logical_ name
```

explain

- Expr is a column value or an expression based on a column value that returns an integer value;
- value_ A list is a list of integers separated by commas.
- The types of columns and the functions and operators supported by expressions refer to the specific descriptions in the overview

Example

Example 1: Create LIST partition table.

```
gbase> create table t1 (a int, b varchar(10))
```

```
        partition by list(a) (
            partition p0 values in (3,5,6,9,17),
            partition p1 values in (1,2,10,11,19,20),
            partition p2 values in (4,12,13,14,18),
            partition p3 values in (7,8,15,16)
        );
Query OK, 0 rows affected (Elapsed: 00:00:00.11)
```

Example 2: Create a LIST partition table with hash sub partitions and no sub partition name specified.

```
gbase> create table t1 (a int, b int)
        partition by list(a)
        subpartition by hash (b)
        subpartitions 4
        (
            partition p0 values in (3,5,6,9,17),
            partition p1 values in (1,2,10,11,19,20),
            partition p2 values in (4,12,13,14,18)
        );
Query OK, 0 rows affected (Elapsed: 00:00:00.16)
```

Example 3: Create a LIST partition table with hash sub partitions and specify the sub partition name.

```
gbase> create table t1 (a int, b int)
        partition by list(a)
        subpartition by hash (b)
        (
        partition p0 values in (3,5,6,9,17)
        (
```

```
        subpartition part0_ q1,
        subpartition part0_ q2,
        subpartition part0_ q3,
        subpartition part0_ q4
        ),
        partition p1 values in (1,2,10,11,19,20)
        (
        subpartition part1_ q1,
        subpartition part1_ q2,
        subpartition part1_ q3,
        subpartition part1_ q4
        ),
        partition p2 values in (4,12,13,14,18)
        (
        subpartition part2_ q1,
        subpartition part2_ q2,
        subpartition part2_ q3,
        subpartition part2_ q4
        )
        );
Query OK, 0 rows affected (Elapsed: 00:00:00.16)
```

Example 4: Create a LIST partition table with key sub partitions and no sub partition name specified.

```
gbase> create table t1 (a int, b int)
        partition by list(a)
        subpartition by key (b)
        subpartitions 4
```

```
        (
            partition p0 values in (3,5,6,9,17),
            partition p1 values in (1,2,10,11,19,20),
            partition p2 values in (4,12,13,14,18)
        );
Query OK, 0 rows affected (Elapsed: 00:00:00.16)
```

Example 5: Create a LIST partition table with a key sub partition and specify the sub partition name.

```
gbase> create table t1 (a int, b int)
        partition by list(a)
        subpartition by key (b)
        (
        partition p0 values in (3,5,6,9,17)
        (
        subpartition part0_ q1,
        subpartition part0_ q2,
        subpartition part0_ q3,
        subpartition part0_ q4
        ),
        partition p1 values in (1,2,10,11,19,20)
        (
        subpartition part1_ q1,
        subpartition part1_ q2,
        subpartition part1_ q3,
        subpartition part1_ q4
        ),
        partition p2 values in (4,12,13,14,18)
```

```
(
subpartition part2_ q1,
subpartition part2_ q2,
subpartition part2_ q3,
subpartition part2_ q4
)
);
Query OK, 0 rows affected (Elapsed: 00:00:00.18)
```

5.1.8.2.1.4.4Create HASH partition

Grammar format

```
partition_ options:
PARTITION BY [LINEAR] HASH(expr)
[PARTITIONS num]
[(partition_definition [, partition_definition] ...)]
partition_ definition:
    PARTITION partition_ name
```

explain

- Expr is a column value or an expression based on a column value that returns an integer value;
- When creating a partition table, num is greater than 8192, and an error is reported;
- In the process of creating a partition table, num equals 0, and an error is reported.
- The types of columns and the functions and operators supported by expressions refer to the specific descriptions in the overview.

Example

Example 1: Creating a hash partition table

```
gbase> create table t1 (a int, b varchar(10)) partition by hash(a);
```

```
Query OK, 0 rows affected (Elapsed: 00:00:00.09)
```

5.1.8.2.1.4.5Create KEY partition

Grammar format

```
partition_ options:
    PARTITION BY [LINEAR] KEY(column_list)
    [PARTITIONS num]
    [(partition_definition [, partition_definition] ...)]
partition_ definition:
     PARTITION partition_ name
```

 **explain**

- column_ A list is a list that uses one or more column names;
- When creating a partition table, num is greater than 8192, and an error is reported;
- In the process of creating a partition table, num equals 0, and an error is reported.
- Please refer to the specific description in the overview for the support of column types

Example

Example 1: Creating a key partition table

```
gbase> create table t1 (a int, b varchar(10))partition by key(a,b) partitions 10;
Query OK, 0 rows affected (Elapsed: 00:00:00.08)
```

5.1.8.2.1.4.6Creating a partition table supports distribution and replication attributes

Example

Example 1: Creating a partition table of a hash distribution table

```
gbase> create table t1 (a int, b varchar(10)) distributed by ('a') partition by
key(a);
```

```
Query OK, 0 rows affected (Elapsed: 00:00:00.09)
```

Example 2: Creating a partition table of a replicated table

```
gbase> create table t2 (a int, b varchar(10))replicated partition by key(a);

Query OK, 0 rows affected (Elapsed: 00:00:00.10)
```

5.1.8.2.1.5 Multi column hash table creation

Function Description

When CREATE TABLE, multiple columns can be specified as HASH columns.

Grammar format

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] [vc_name.][database_name.]table_ name
(column_definition [,column_definition], ... [, key_options])
[table_options]
[NOCOPIES];

table_ options:
[REPLICATED | DISTRIBUTED BY ('column_name','column_name1',...) ]
[COMMENT 'comment_value']
```

Example

```
gbase> create table x0( entry_id int, id2 int, id3 int,id4 int ) distributed by('id3','id4');
Query OK, 0 rows affected (Elapsed: 00:00:00.08)
```

5.1.8.2.2 ALTER TABLE

Grammar format

```
ALTER TABLE [vc_name.][database_name.]table_ name
    alter_ specification [, alter_specification] ...
alter_ specification:
  ADD [COLUMN] column_ definition [FIRST | AFTER col_name ]
 | ADD [COLUMN] (column_definition,...)
 | CHANGE [COLUMN] old_ col_ name new_ col_ name column_ definition
 | MODIFY [COLUMN] col_ name column_ definition
        [FIRST | AFTER col_name]
 | RENAME [TO] new_ table_ name
 | DROP [COLUMN] col_ name
```

Table -556 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional.
database_ name	Database name, optional.
ADD [COLUMN] (column_definition,...)	Used to add new data columns. If FIRST is used, the newly added columns will be in front of all data columns; If after is used, the newly added column will be located after the specified data column. If FIRST and AFTER are not used by default, the newly added columns will be appended to the end.
CHANGE [COLUMN] old_ col_ name new_ col_ name column_ definition	Modify the column name. Modifying column definitions is not supported.
MODIFY [COLUMN] col_ name column_ definition FIRST | AFTER col_ name	Modify the position of columns in the table. Modifying column definitions is not supported.
RENAME [TO] new_ table_ name	Change the table name to new_ table_ name。
DROP [COLUMN] col_ name	Delete columns that exist in the table.

be careful

- Currently, it supports adding columns, deleting columns, changing table names, changing column names, and changing column positions;
- Unsupported options include ORDER BY, changing the data type of a column, changing the attributes of a column (NOT NULL, default value), and changing the character set of a table;
- The restrictions for adding new columns include:
  - For newly added columns, if NOT NULL is set, the default value needs to be set at the same time, otherwise an error will be reported;
  - Non EXPRESS engine tables are not allowed to convert to and from EXPRESS tables.

Example

Example 1: Adding columns:

```
gbase> DROP TABLE IF EXISTS t;
Query OK, 0 rows affected

gbase> CREATE TABLE t (a int NOT NULL DEFAULT 1, b varchar(10));
Query OK, 0 rows affected

gbase> DESC t;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | NO   |     | 1       |       |
| b     | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set

gbase> ALTER TABLE t ADD column c varchar(10) null;
Query OK, 0 rows affected
Records: 0  Duplicates: 0  Warnings: 0

gbase> DESC t;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | NO   |     | 1       |       |
| b     | varchar(10) | YES  |     | NULL    |       |
| c     | varchar(10) | YES  |     | NULL    |       |
```

```
+-------+-------------+------+-----+---------+-------+
3 rows in set
```

Example 2: Delete a column.

```
gbase> DROP TABLE IF EXISTS t;
Query OK, 0 rows affected

gbase> CREATE TABLE t (a int NOT NULL DEFAULT 1, b varchar(10));
Query OK, 0 rows affected

gbase> DESC t;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | NO   |     | 1       |       |
| b     | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set

gbase> ALTER TABLE t DROP column b;
Query OK, 0 rows affected
Records: 0  Duplicates: 0  Warnings: 0

gbase> DESC t;
+-------+---------+------+-----+---------+-------+
| Field | Type    | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| a     | int(11) | NO   |     | 1       |       |
+-------+---------+------+-----+---------+-------+
1 row in set
```

Example 3: Changing Table Names.

```
gbase> SHOW TABLES;
+--------------+
| Tables_ in_ ty |
+--------------+
| t            |
+--------------+
1 row in set

gbase> ALTER TABLE t RENAME ttt2;
Query OK, 0 rows affected

gbase> SHOW TABLES;
```

```
+--------------+
| Tables_ in_ ty |
+--------------+
| ttt2           |
+--------------+
1 row in set
```

Example 4: Changing column name b to a new column name d.

```
gbase> CREATE TABLE t (a int not null,b varchar(10),c varchar(10));
Query OK, 0 rows affected

gbase> DESC t;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | NO   |     | NULL    |       |
| b     | varchar(10) | YES  |     | NULL    |       |
| c     | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set

gbase> ALTER TABLE t  CHANGE  b  d varchar(10);
Query OK, 0 rows affected
Records: 0  Duplicates: 0  Warnings: 0

gbase> DESC t;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | NO   |     | NULL    |       |
| d     | varchar(10) | YES  |     | NULL    |       |
| c     | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set
```

Example 5: Change the position of the column to the top.

```
gbase> DROP TABLE IF EXISTS t;
Query OK, 0 rows affected

gbase> CREATE TABLE t(a int ,b varchar(10),c bool);
Query OK, 0 rows affected

gbase> DESC t;
+-------+-------------+------+-----+---------+-------+
```

```
| Field | Type              | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | YES  |     | NULL    |       |
| b     | varchar(10) | YES  |     | NULL    |       |
| c     | tinyint(1)  | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set

gbase> ALTER TABLE t   MODIFY b varchar(10) FIRST;
Query OK, 0 rows affected
Records: 0   Duplicates: 0   Warnings: 0

gbase> DESC t;
+-------+-------------+------+-----+---------+-------+
| Field | Type              | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| b     | varchar(10) | YES  |     | NULL    |       |
| a     | int(11)     | YES  |     | NULL    |       |
| c     | tinyint(1)  | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set
```

Example 6: Change the position of a column to the end of the specified column.

```
gbase> ALTER TABLE t MODIFY b varchar(10) AFTER c;
Query OK, 0 rows affected
Records: 0   Duplicates: 0   Warnings: 0

gbase> DESC t;
+-------+-------------+------+-----+---------+-------+
| Field | Type              | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | YES  |     | NULL    |       |
| c     | tinyint(1)  | YES  |     | NULL    |       |
| b     | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set
```

5.1.8.2.2.1 ALTER TABLE... SHRINK SPACE

Function Description

Release the disk space occupied by deleted data files.

- alter table t shrink space full;

FULL: **Line level recycling**, organizing valid data in the DC in the original order of the

line level, and reloading and writing SEG files. As long as there is deleted data, the space for deleted data will be reclaimed. The order of effective rows remains consistent with before recycling, resulting in lower efficiency.

- alter table t shrink space full block_ reuse_ ratio=30;

FULL BLOCK_ REUSE_ RATIO: **Block level (DC) recycling**. When the proportion of valid data in the DC is greater than or equal to the set value, the DC is retained for reuse. When the proportion of valid data is lower than the set value, the DC space is consolidated to reclaim invalid data space. After the DC is consolidated, it is reloaded and written to the SEG file, which cannot guarantee the original order.

- alter table t shrink space;

Original syntax: **seg level recycling**. If all data within the seg is deleted, the seg file space will be reclaimed.

Grammar format

```
ALTER TABLE [vc_name.][database_name.]table_ name SHRINK SPACE [FULL] | [FULL BLOCK_REUSE_RATIO= num]
```

be careful

- SEG level recycling, does not recycle index files and rowids.
- Row level recycling and block level recycling will recycle index files and rowids.
- The disk space reclamation command is only for tables.
- Tables containing rows and columns do not support row level and block level recycling.
- If the deletion hits all data, files with trailing block data will not be cleaned up.
- During the disk space recycling process, a certain amount of disk space is required to back up some metadata files. Executing this command when there is no available space will result in an error. At this point, it is necessary to manually clean up a portion of the space (usually 1GB of space) and then execute this command for space recycling.
- Rebalance implements the coexistence of the shrink space function and the original shrink space function, using the parameter gcluster_ shrink_ to_ Rebalance control. The default value is 0; Value 1: Turn on the shrink to rebalance function; It can improve the performance of shrink spaces, without blocking DML operations such as insert select and load append only, and supports DQL on tables in rebalance. Please refer to the parameter configuration section for specific instructions on this parameter.

Example

Example 1: Release the disk space occupied by deleted data files.

```
gbase> select count(*) from lineorder;
+----------+
| count(*) |
+----------+
|  6001215 |
+----------+
1 row in set (Elapsed: 00:00:00.41)

[ root@rhel73-1  lineorder_ n1]# pwd
/opt/gnode/userdata/gbase/ssbm/sys_ tablespace/lineorder_ n1
[ root@rhel73-1  lineorder_ n1]# ll -h
Total usage 69M
-Rw ------1 gbase gbase 1.6M September 28th 11:32 C00000.seg.1
-Rw ------1 gbase gbase 734K September 28th 11:32 C00001. seg.1
-Rw ------1 gbase gbase 1.6M September 28th 11:32 C00002. seg.1
-Rw ------1 gbase gbase 5.8M September 28th 11:32 C00003. seg.1
-Rw ------1 gbase gbase 2.9M September 28th 11:32 C00004. seg.1
-Rw ------1 gbase gbase 24K September 28th 11:32 C00005. seg.1
-Rw ------1 gbase gbase 2.8M September 28th 11:32 C00006.seg.1
-Rw ------1 gbase gbase 31K September 28th 11:32 C00007. seg.1
-Rw ------1 gbase gbase 1.5M September 28th 11:32 C00008. seg.1
-Rw ------1 gbase gbase 5.8M September 28th 11:32 C00009.seg.1
-Rw ------1 gbase gbase 2.4M September 28th 11:32 C00010. seg.1
-Rw ------1 gbase gbase 1.2M September 28th 11:32 C00011. seg.1
-Rw ------1 gbase gbase 5.8M September 28th 11:32 C00012. seg.1
-Rw ------1 gbase gbase 5.3M September 28th 11:32 C00013. seg.1
-Rw ------1 gbase gbase 1.1M September 28th 11:32 C00014.seg.1
-Rw ------1 gbase gbase 2.2M September 28th 11:32 C00015.seg.1
-Rw ------1 gbase gbase 6.0M September 28th 11:32 C00016.seg.1

gbase> delete from lineorder where lo_ orderkey<=5000000;
Query OK, 5001154 rows affected (Elapsed: 00:00:02.10)

gbase> select count(*) from lineorder;
+----------+
| count(*) |
+----------+
|  1000061 |
+----------+
1 row in set (Elapsed: 00:00:00.00

gbase> alter table lineorder shrink space full;
Query OK, 0 rows affected (Elapsed: 00:00:06.53)
```

```
[ root@rhel73-1    lineorder_ n1]# ll -h
Total usage 12M
-Rw ------1 gbase gbase 274K September 28th 11:35 C00000.seg.1. B
-Rw ------1 gbase gbase 123K September 28th 11:35 C00001. seg.1. B
-Rw ------1 gbase gbase 268K September 28th 11:35 C00002. seg.1. B
-Rw ------1 gbase gbase 978K September 28th 11:35 C00003. seg.1. B
-Rw ------1 gbase gbase 489K September 28th 11:35 C00004. seg.1. B
-Rw ------1 gbase gbase 4.0K September 28th 11:35 C00005. seg.1. B
-Rw ------1 gbase gbase 479K September 28th 11:35 C00006. seg.1. B
-Rw ------1 gbase gbase 5.2K September 28th 11:35 C00007.seg.1. B
-Rw ------1 gbase gbase 245K September 28th 11:35 C00008. seg.1. B
-Rw ------1 gbase gbase 978K September 28th 11:35 C00009.seg.1. B
-Rw ------1 gbase gbase 394K September 28th 11:35 C00010. seg.1. B
-Rw ------1 gbase gbase 199K September 28 11:35 C00011. seg.1. B
-Rw ------1 gbase gbase 978K September 28th 11:35 C00012. seg.1. B
-Rw ------1 gbase gbase 891K September 28th 11:35 C00013. seg.1. B
-Rw ------1 gbase gbase 186K September 28th 11:35 C00014.seg.1. B
-Rw ------1 gbase gbase 320K September 28th 11:35 C00015.seg.1. B
-Rw ------1 gbase gbase 1021K September 28th 11:35 C00016. seg.1. B
```

5.1.8.2.2.2 ALTER TABLE…ADD PARTITION

Grammar format

```
ALTER TABLE [vc_name.][database_names.]table_ name

    alter_ specification [, alter_specification] ...

    alter_ specification:

    ADD PARTITION partition_ definition
```

explain

- The partition condition of the partition table is RANGE or LIST;
- For a RANGE partition table, the value range of the new partition must be incremental compared with the value range of the existing partition in the table (LESS THAN MAXVALUE is the last partition, and no partition can be added after MAXVALUE);
- For LIST partition table, the new partition cannot contain any value in the existing partition value list;
- The partition table does not allow adding child partitions.

Example

```
gbase> create table pt (i int ,c varchar(10),d date)
        partition by range(i) (
        partition p0 values less than(10),
        partition p1 values less than(20));
Query OK, 0 rows affected (Elapsed: 00:00:00.13)

Gbase>- Add partitions
gbase> alter table pt add partition (partition p2 values less than(30));
Query OK, 0 rows affected (Elapsed: 00:00:00.12)
Records: 0  Duplicates: 0  Warnings: 0

gbase> alter table pt add partition (partition p3 values less than(maxvalue));
Query OK, 0 rows affected (Elapsed: 00:00:00.13)
Records: 0  Duplicates: 0  Warnings: 0
```

be careful

- When adding a partition, the partition conditions do not meet the rules and an error is reported;
- When adding a partition, the partition name is duplicated and an error is reported;
- When adding partitions, if the number of partitions exceeds 8192, an error will be reported;
- When adding a partition, the partition name does not comply with naming conventions and an error is reported;
- When adding a partition, if it is a non RANGE or LIST partition, an error will be reported.

5.1.8.2.2.3 ALTER TABLE…DROP PARTITION

Grammar format

```
ALTER TABLE [vc_name.][database_names.]tbl_ name
    alter_ specification [, alter_specification] ...
   alter_ specification:
    DROP PARTITION partition_ names
```

explain

- The partition condition of the partition table is RANGE or LIST;
- Cannot delete all partitions, ensure that at least one partition exists;
- Cannot delete sub partitions.

Example

```
gbase> show create table pt\G
*************************** 1. row ***************************
       Table: pt
Create Table: CREATE TABLE "pt" (
  "i" int(11) DEFAULT NULL,
  "c" varchar(10) DEFAULT NULL,
  "d" date DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
 PARTITION BY RANGE (i)
(PARTITION p0 VALUES LESS THAN (10) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS,
 PARTITION p1 VALUES LESS THAN (20) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS,
 PARTITION p2 VALUES LESS THAN (30) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS,
 PARTITION p3 VALUES LESS THAN MAXVALUE TABLESPACE = 'sys_
tablespace' ENGINE = EXPRESS)
1 row in set (Elapsed: 00:00:00.00)

gbase> alter table pt drop partition p3;
Query OK, 0 rows affected (Elapsed: 00:00:00.11)
Records: 0  Duplicates: 0  Warnings: 0

gbase> alter table pt drop partition p1;
Query OK, 0 rows affected (Elapsed: 00:00:00.12)
Records: 0  Duplicates: 0  Warnings: 0
```

```
gbase> show create table pt\G
*************************** 1. row ***************************
       Table: pt
Create Table: CREATE TABLE "pt" (
  "i" int(11) DEFAULT NULL,
  "c" varchar(10) DEFAULT NULL,
  "d" date DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
 PARTITION BY RANGE (i)
(PARTITION p0 VALUES LESS THAN (10) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS,
 PARTITION p2 VALUES LESS THAN (30) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS)
1 row in set (Elapsed: 00:00:00.01)
```

be careful

- When deleting a partition table partition, if there is only one partition, an error is reported;
- When deleting partition table partitions, if all partitions are deleted at the same time, an error is reported;
- When deleting a partition table partition, if it is not a RANGE or LIST partition, an error is reported.

5.1.8.2.2.4 ALTER TABLE…TRUNCATE PARTITION

Grammar format

```
ALTER TABLE table_ name TRUNCATE PARTITION partition_ name1;
```

Table -597 Parameter Description

Parameter Name	explain
table_ name	The name of the created partition table.
partition_ name1	Specify the partition to clear.

explain

- Delete all data within the partition, but retain the partition structure information;
- Truncate of sub partitions is not supported.

Example

Example 1: Projection columns use alias dependencies.

```
create table t5 (a int, b varchar(10))
partition by range(a)(
        partition p0 values less than(10),
        partition p1 values less than(20),
        partition p2 values less than(30),
        partition p3 values less than(40)
);

Query OK, 0 rows affected (Elapsed: 00:00:00.25)

insert into t5 values (31,'wwd'),(32,'aas'),(33,'aaw'),(34,'aaz');

Query OK, 4 rows affected (Elapsed: 00:00:00.08)
Records: 4    Duplicates: 0    Warnings: 0

select * from t5;

+-------+-------+
| a       | b       |
+-------+-------+
|     31 | wwd    |
|     32 | aas    |
|     33 | aaw    |
|     34 | aaz    |
+-------+-------+
4 rows in set (Elapsed: 00:00:00.03)

alter table t5 truncate partition p3;

Query OK, 0 rows affected (Elapsed: 00:00:02.27)

select * from t5;

Empty set (Elapsed: 00:00:00.03)
```

```
insert into t5 values (11,'wwd'),(22,'aas'),(33,'aaw'),(4,'aaz');

select * from t5;

+-------+-------+
| a     | b     |
+-------+-------+
|     4 | aaz   |
|    11 | wwd   |
|    22 | aas   |
|    33 | aaw   |
+-------+-------+
4 rows in set (Elapsed: 00:00:00.02)

alter table t5 truncate partition p3;

Query OK, 0 rows affected (Elapsed: 00:00:00.10)

select * from t5;

+-------+-------+
| a     | b     |
+-------+-------+
|     4 | aaz   |
|    11 | wwd   |
|    22 | aas   |
+-------+-------+
3 rows in set (Elapsed: 00:00:00.03)
```

5.1.8.2.2.5 ALTER TABLE…RENAME PARTITION

Grammar format

ALTER TABLE *table_ name* RENAME PARTITION *partition_ name1* to new_ partition_ name;

Table 597 Parameter Description

Parameter Name	explain
table_ name	The name of the created partition table.
partition_ name1	The old name of the original partition.

Parameter Name	explain
new_ partition_ name	The new name of the original partition.

explain

- Change the name information of the partition and retain both data and partition structure information;
- Rename of sub partitions is not supported.

Example

Example 1: Projection columns use alias dependencies.

```
create table t6 (a int, b varchar(10))
partition by range(a)(
        partition p0 values less than(10),
        partition p1 values less than(20),
        partition p2 values less than(30),
        partition p3 values less than(40),
);

Query OK, 0 rows affected (Elapsed: 00:00:00.23)

alter table t6 rename partition p3 to p4;

Query OK, 0 rows affected (Elapsed: 00:00:00.35)

show create table t6;

+-------+-----------------------------------------------------------------------------------------+
| Table | Create Table
|
+-------+-----------------------------------------------------------------------------------------+
| t6       | CREATE TABLE "t6" (
   "a" int(11) DEFAULT NULL,
   "b" varchar(10) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
  PARTITION BY RANGE (a)
(PARTITION p0 VALUES LESS THAN (10) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS,
  PARTITION p1 VALUES LESS THAN (20) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS,
```

```
 PARTITION p2 VALUES LESS THAN (30) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS,
 PARTITION p4 VALUES LESS THAN (40) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS) |
+-------+----------------------------------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

alter table t6 rename partition p4 to p3;

Query OK, 0 rows affected (Elapsed: 00:00:00.17)

show create table t6;

+-------+----------------------------------------------------------------------------------------+
| Table | Create Table
|
+-------+----------------------------------------------------------------------------------------+
| t6    | CREATE TABLE "t6" (
  "a" int(11) DEFAULT NULL,
  "b" varchar(10) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
 PARTITION BY RANGE (a)
(PARTITION p0 VALUES LESS THAN (10) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS,
 PARTITION p1 VALUES LESS THAN (20) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS,
 PARTITION p2 VALUES LESS THAN (30) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS,
 PARTITION p3 VALUES LESS THAN (40) TABLESPACE = 'sys_ tablespace'
ENGINE = EXPRESS) |
+-------+----------------------------------------------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.8.2.3 **TRUNCATE TABLE**

Function Description

TRUNCATE TABLE is functionally the same as a DELETE statement without a WHERE clause, both of which delete all rows in the table. But TRUNCATE TABLE is faster than DELETE and uses fewer system and transaction log resources.

TRUNCATE TABLE belongs to DDL syntax, DELETE From table_ Name belongs to DML syntax.

TRUNCATE TABLE deletes all rows in the table, but the table structure and its columns, constraints, indexes, etc. remain unchanged.

Grammar format

```
TRUNCATE TABLE [vc_name.][database_name.]table_ name
```

Table -559 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional;
database_ name	Is the name of the database to which the table belongs to be deleted, optional; Omitting this parameter means the USE database_ The database name after name.
table_ name	Is the name of the table to delete all rows from.

Example

Example 1: Delete all data in table t.

```
gbase> USE test;
Query OK, 0 rows affected

gbase> CREATE TABLE t (a decimal(12,5) DEFAULT NULL, KEY idx_ a (a) USING HASH global);
Query OK, 0 rows affected

gbase> INSERT INTO t VALUES(1),(2),(3);
Query OK, 3 rows affected
Records: 3  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t;
+---------+
| a       |
+---------+
| 1.00000 |
| 2.00000 |
| 3.00000 |
+---------+
3 rows in set

gbase> TRUNCATE TABLE t;
Query OK, 3 rows affected

gbase> SELECT * FROM t;
```

```
Empty set
```

5.1.8.2.4 DROP TABLE

Grammar format

```
DROP [TEMPORARY] TABLE [IF EXISTS] [vc_name.][database_name.]table_ name;
```

Table 560 Parameter Description

Parameter Name	explain
TEMPORARY	This parameter is optional, and it is recommended to use this keyword when deleting temporary tables.
IF EXISTS	Users can use the keyword IF EXISTS to prevent reporting errors when the table does not exist. When using IF EXISTS, users will receive a warning for non-existent tables.
vc_ name	Virtual cluster name, optional;
database_ name	Is the name of the database to which the table belongs to be deleted, optional; Omitting this parameter means the USE database_ The database name after name.
table_ name	Is the name of the table to delete all rows from.

be careful

When using DROP TABLE to remove a table, all data and table definitions will be deleted. Users must have DROP permission on the table, so be careful when using this command!

5.1.8.3 TABLE SPACE

GBase 8a MPP Cluster: A tablespace represents a data storage path. The rule for creating tablespaces is that each library can have multiple tablespaces, with only one default tablespace. A tablespace can be used by multiple tables, but a table can only belong to one tablespace.

By default, each library has a default tablespace sys_ tablespace， sys_ The tablespace

points to the current fixed data storage path (in the sys_tablespace directory under the database name directory specified by datadir in $GBASE_BASE/config/gbase_8a_gbase.cnf) and cannot be deleted.

5.1.8.3.1 Specify default tablespace when building a library

Grammar format

```
CREATE DATABASE [vc_name.]<database_ name> SYSTEM TABLESPAC
E DATADIR <path> [SEGSIZE <segsize_value>] [MAXSIZE <max size_va
lue>];
```

Table 561 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional.
database_ name	Database name, optional.
path	The default Tablespace corresponds to the system path, which supports both relative and absolute paths. The relative path is relative to the datadir path configured in the configuration file.
segsize_ value	The size of seg file splitting for tables in the tablespace is between 10M and 2G, expressed in units with K, M, and G, and defaults to 2G.
maxsize_ value	Specify the maximum table space limit, which must be greater than the segsize value, expressed in units with K, M, and G. Ensure that the available disk space is greater than maxsize by the user_ value。 The default is unrestricted. This parameter, when specified at the cluster level, represents the maximum table space limit for each node. Resource management precedes table space for space limit checking.

5.1.8.3.2 CREATE TABLESPACE

Grammar format

```
CREATE TABLESPACE [IF NOT EXISTS] [[vc_name.]database_ name.]< t
```

ablespace_ name> DATADIR <*path*> [SEGSIZE <*segsize_value*>] [MAXSIZ E <*maxsize_value*>]

Table 562 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional.
database_ name	Database name, optional.
tablespace_ name	The tablespace name supports numbers in English and underscores, with a length of 64 characters and is not case sensitive. It is created entirely using lowercase characters and does not allow numbers to start. (This rule applies to the entire text, and similar content will not be repeated);
path	The actual system path corresponding to the Tablespace, which supports both relative and absolute paths. The relative path is relative to the datadir path configured in the configuration file;
segsize_ value	The size of seg file splitting for tables in Tablespace, between 10M and 2G, expressed in units of K, M, and G, defaults to 2G;
maxsize_ value	Specify the maximum table space limit, which must be greater than the segsize value, expressed in units with K, M, and G. Ensure that the available disk space is greater than maxsize by the user_ value。 The default is unrestricted. This parameter, when specified at the cluster level, represents the maximum table space limit for each node. Resource management precedes table space for space limit checking.

5.1.8.3.3 DROP TABLESPACE

Grammar format

DROP TABLESPACE [*vc_name.*]database_ *name.*]< *tablespace_ name*>;

Table 563 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional.
database_ name	Database name, optional.

Parameter Name	explain
tablespace_ name	Tablespace name.

be careful

The following restrictions apply to deleting tablespaces:

- The default tablespace cannot be deleted;
- System tablespace sys_ Tablespace cannot be deleted;
- The tablespace in use cannot be deleted.

5.1.8.3.4 ALTER TABLESPACE

Function Description

Modify the TABLESPACE maximum limit.

Grammar format

```
ALTER TABLESPACE [[vc_name.]database_ name.] tablespace_ name MAXSIZE <maxsize_ value>;
```

Table 564 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional.
database_ name	Database name, optional.
tablespace_ name	Tablespace name.
maxsize_ value	Specify the maximum table space limit, which must be greater than the segsize value, expressed in units of K, M, and G. Ensure that the available disk space is greater than maxsize by the user_ value 。 The default is unrestricted. Only increase and modification are supported, and reduction is not allowed. Resource management precedes table space for space limit checking.

5.1.8.3.5 Specify tablespace when creating a table

Function Description

When creating a table, you can specify the TABLESPACE to use, otherwise the default tablespace of the library will be used. Partition table can only be specified as one table. It is not supported to specify different tablespaces for different partitions.

Grammar format

```
CREATE TABLE [[vc_name.]database_ name.] table_ name (column_def...)
TABLESPASCE=tablespace_ name;
```

Table 565 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional.
database_ name	Database name, optional.
table_ name	Table name.
tablespace_ name	Tablespace name.

explain

- CREATE TABLE... LIKE... does not support specifying tablespaces;
- CREATE TABLE... AS SELECT... supports specifying tablespaces.

5.1.8.3.6 Set the default tablespace for the library

Function Description

Users can modify the default Tablespace of the specified library. After setting the default Tablespace of the database, when creating a table without specifying a Tablespace, the default Tablespace will be used. This feature does not support system library operations, including gbase, gctmpdb, and performance_ Schema and information_ Schema, etc.

Grammar format

```
USE [[vc_name.]database_ name.] tablespace_ name AS DEFAULT TABLE
SPACE;
```

Table 566 Parameter Description

Parameter Name	explain

Parameter Name	explain
vc_ name	Virtual cluster name, optional.
database_ name	Database name, optional.
tablespace_ name	Tablespace name.

explain

Before using use, it is necessary to ensure that the tablespace to be set has been created.

5.1.8.3.7 **SHOW TABLESPACES**

Function Description

Query tablespace information.

Grammar format

```
SHOW [full] TABLESPACES;
```

Table 567 Parameter Description

Parameter Name	explain
full	Display whether it is the default tablespace.

Example

```
gbase> SHOW FULL TABLESPACES;
+------------------------+------------------------+------------+
| Tablespace_ in_ test_ sdy | Tablespace_ in_ test_ sdy | Is_ default |
+------------------------+------------------------+------------+
| sys_ tablespace | . | no |
| tbs1 | ../ tbs1 | yes |
+------------------------+------------------------+------------+
2 rows in set (Elapsed: 00:00:00.00)
```

5.1.8.3.8 REFRESH TABLESPACE

Function Description

Refresh tablespace usage size information.

Grammar format

```
REFRESH TABLESPACE [[vc_name.]database_ name.] tablespace_ nameST
ORAGE USAGE
```

Table 568 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional.
database_ name	Database name, optional.
tablespace_ name	Tablespace name.

be careful

- Does not support TABLESPACE related operations under the gctmpdb library and gclusterdb library, except for SHOW TABLESPACES;
- Before establishing an image relationship, users need to manually create the specified directory and corresponding TABLESPACE on the relevant nodes in advance, and ensure their consistency. After the image relationship is established, it supports the image issuance function of creating the TABLESPACE command;
- When uninstalling, the data in the relevant directories of the user-defined tablespace will be deleted;
- There are three levels of backup and recovery tools: instance level, library level, and table level. Among them, the instance level and library level recovery processes are completed by the program, and users do not need to manually create TABLESPACE and corresponding directories; The table level recovery process requires users to manually establish the same TABLESPACE in the specified library in advance as during backup;
- Regarding the expansion function, users need to manually establish the directory required for TABLESPACE on the newly expanded node in advance, and the action of creating TABLESPACE on the new node is completed by the program itself;
- The partition table is treated as one table. The user can only specify one TABLESPACE. Specifying different TABLESPACE for different partitions is not supported;
- For the node replacement function, users need to manually establish the TABLESPACE path on the replaced node in advance.

5.1.8.4 VIEW

A view is a virtual table that, like a real table, contains a series of named columns. The data of the view comes from the table referenced by the query that defines the view, and is dynamically generated when referencing the view. For the underlying table referenced therein, the function of a view is similar to filtering. The filtering of defined views can come from one or more tables in the current or other databases, or from other views.

Views have the following functions:

1）Simplicity: What you see is what you need. Views can not only simplify users' understanding of data, but also simplify their operations. The frequently used queries can be defined as views, so that users do not have to specify all the conditions for future operations each time.

2）Security: Users can only query the data they can see through the view. The other data in the database is neither visible nor accessible. The database authorization command can restrict each user's retrieval of the database to specific database objects, but cannot be authorized to specific database rows. Through views, users can be limited to different subsets of data.

be careful

- GBase 8a MPP Cluster prohibits Insert, UPDATE, and DELETE operations on views.

5.1.8.4.1 CREATE VIEW

Function Description

This statement is used to create a new view, or to modify an existing view definition using the OR REPLACE clause.

explain

- Creating a view requires CREATE VIEW permission, as well as partial permissions for referencing columns in the SELECT statement that makes up the view. The columns to be used in a SELECT statement must have SELECT permission. If the OR Replace clause is used, you must also have permission to delete the view;
- Within stored subroutines, definitions cannot reference subroutine parameters or local variables;
- The table or view referenced in the definition must exist. However, after creating a view, it is possible to discard the tables or views that define references. To check whether the view definition has such issues, you can use the CHECK TABLE statement;
- The table named in the view definition must already exist.
- Cannot associate trigger programs with views;
- The TEMPORARY table cannot be referenced in the view definition;
- The SELECT statement cannot contain subqueries in the From clause;
- The SELECT statement cannot reference system or user variables;
- The SELECT statement cannot reference preprocessed statement parameters;
- The SELECT statement field name can be followed by a comment, with a maximum comment character limit of 2000;
- The created view name cannot have the same name as an existing table.

Grammar format

```
CREATE [OR REPLACE] VIEW [vc_name.][database_name.]view_ name
[(column_list)] AS select_ statement;
```

Table 569 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
view_ name	View name.
column_ list	View columns.
select_ statement	Provide a SELECT statement to define the view. This statement can extract data from other tables or views, and supports adding fields to the view and adding annotations to the view fields. select select_ expr [comment comment_value],… from table_ references

Example

Example 1: Create a view.

```
gbase> DROP TABLE IF EXISTS product;
Query OK, 0 rows affected

gbase> CREATE TABLE product (quantity INT,price INT);
Query OK, 0 rows affected

gbase> INSERT INTO product VALUES(3,50);
Query OK, 1 row affected

gbase> CREATE VIEW product_ v AS SELECT quantity,price,quantity*price FROM product;
Query OK, 0 rows affected

gbase> SELECT * FROM product_ v;
+----------+-------+-----------------+
| quantity | price | quantity*price  |
+----------+-------+-----------------+
|        3 |    50 |             150 |
+----------+-------+-----------------+
1 row in set

Create annotated views:
Create view v_ user as select id comment 'user id', addr comment 'user office address' from user;
Create view v_ user as select id as 'id' comment 'user id', addr as 'addr' comment 'user office address' from user;
```

be careful

- The length limit for view annotation content is the same as the annotation length limit for fields in the table, both up to a maximum of 2000 bytes.
- How to view view field annotations:
  Show create view view name;
  Show create table view name;

5.1.8.4.2 ALTER VIEW

Function Description

Modify the view column fields.

Grammar format

```
ALTER VIEW [vc_name.][database_name.]view_ name [(column_list)] AS
select_ statement
```

Table 570 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
view_ name	View name.
column_ list	List of modified view columns.
select_ statement	Provide a SELECT statement to define the view. This statement can extract data from other tables or views.

Example

Example 1: Modifying View v_ The columns in t are the specified columns.

```
gbase> CREATE TABLE t (name VARCHAR(20),address VARCHAR(40),sex INT);
Query OK, 0 rows affected (Elapsed: 00:00:00.06)

gbase> CREATE VIEW v_ t AS SELECT * FROM t;
Query OK, 0 rows affected (Elapsed: 00:00:00.05)

gbase> INSERT INTO t VALUES('TOM','east street','23'),('jack','west road NO 15','22'),('MIKE','DongFang road NO 22','21'),('TONY','EA Street','34'),('Rose','TangRen Street NO.191','31');
Query OK, 5 rows affected (Elapsed: 00:00:00.05)
Records: 5  Duplicates: 0  Warnings: 0

gbase> DESC v_ t;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| name    | varchar(20) | YES  |     | NULL    |       |
| address | varchar(40) | YES  |     | NULL    |       |
| sex     | int(11)     | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
3 rows in set (Elapsed: 00:00:00.01)
```

```
gbase> SELECT * FROM v_ t;
+------+----------------------+------+
| name | address              | sex  |
+------+----------------------+------+
| TOM  | east street          |   23 |
| jack | west road NO 15      |   22 |
| MIKE | DongFang road NO 22  |   21 |
| TONY | EA Street            |   34 |
| Rose | TangRen Street NO.191 |   31 |
+------+----------------------+------+
5 rows in set (Elapsed: 00:00:00.03)

gbase> ALTER VIEW v_ t(a,b) AS SELECT name,address FROM t;
Query OK, 0 rows affected (Elapsed: 00:00:00.07)

gbase> DESC v_ t;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | varchar(20) | YES  |     | NULL    |       |
| b     | varchar(40) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (Elapsed: 00:00:00.00)

gbase> SELECT * FROM v_ t;
+------+----------------------+
| a    | b                    |
+------+----------------------+
| TOM  | east street          |
| jack | west road NO 15      |
| MIKE | DongFang road NO 22  |
| TONY | EA Street            |
| Rose | TangRen Street NO.191 |
+------+----------------------+
5 rows in set (Elapsed: 00:00:00.02)
```

5.1.8.4.3 DROP VIEW

Grammar format

```
DROP VIEW [IF EXISTS] [vc_name.][database_name.]view_ name;
```

Table 571 Parameter Description

Parameter Name	explain
IF EXISTS	Prevent reporting errors when the view does not exist.
vc_ name	VC name, optional.
database_ name	Database name, optional.

explain

- DROP VIEW deletes a view. The user must have DROP permission on the view.
- DROP VIEW can only delete one view at a time.

Example

Example 1: To delete a single view:

```
gbase> DROP VIEW IF EXISTS student_ v;
Query OK, 0 rows affected
```

5.1.8.5 INDEX

5.1.8.5.1 CREATE INDEX

Grammar format

```
CREATE INDEX index_ name ON [vc_name.][database_name.]table_ name (column_name) [key_block_size = size_value] [key_dc_size=num] USING HASH GLOBAL;
```

Table 572 Parameter Description

Parameter Name	explain
index_ name	Index name.
vc_ name	VC name, optional.
database_ name	Database name. Can be omitted. When omitted, the USE command must be used first to specify the current database.
column_ name	The column name for creating the index.
GLOBAL	Index type, which can be omitted. If omitted, a GLOBAL hash index is created by default.

Parameter Name	explain
key_ dc_ size=num	How many DataCells create a hash index. When this parameter is specified, it indicates that a segmented hash index is being created.
key_ block_ size = size_ value	Specify the size of each data block. Size_ Minimum value 4096, maximum value 32768, size_ Value must be an integer multiple of 4096.

explain

- Binary type columns are not suitable for using HASH INDEX, or when the column has a large amount of data but a small DISTINCT value, it is also not suitable for using HASH INDEX.
- Hash indexes with the same name cannot be created on the same table, and only one hash index can be created on the same column of the same table. Hash indexes can be created on columns of any data type supported by GBase 8a MPP Cluster.
- After creating a hash index, the performance of equivalent queries based on index columns will improve, especially when the data volume in the table is very large. In small data volumes, the performance improvement effect of hash indexes is not significant.

Example

Example 1: Create a default hash index.

```
gbase> CREATE TABLE t1(a int,b varchar(10));
Query OK, 0 rows affected

gbase> CREATE INDEX idx1 on t1(a) ;
Query OK, 0 rows affected
Records: 0    Duplicates: 0    Warnings: 0

gbase> SHOW CREATE TABLE t1\G
*************************** 1. row ***************************
           Table: t1
Create Table: CREATE TABLE "t1" (
  "a" int(11) DEFAULT NULL,
```

```
  "b" varchar(10) DEFAULT NULL,
  KEY "idx1" ("a") USING HASH GLOBAL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: Creating a GLOBAL hash index and setting the key_ dc_ The size value is 1000;

```
gbase> CREATE TABLE t (a INT);
Query OK, 0 rows affected

gbase> CREATE INDEX idx_ t_ a ON t(a) key_ DC_ size = 1000 USING
HASH GLOBAL;
Query OK, 0 rows affected
Records: 0  Duplicates: 0  Warnings: 0

gbase> SHOW CREATE TABLE t\G
*************************** 1. row ***************************
       Table: t
Create Table: CREATE TABLE "t" (
  "a" int(11) DEFAULT NULL,
  KEY "idx_t_a" ("a") KEY_ DC_ SIZE=1000 USING HASH GLOBAL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 3: Creating a GLOBAL hash index and setting the key_ block_ The size value is 16384.

```
gbase> CREATE TABLE t3(a int,b varchar(10));
Query OK, 0 rows affected

gbase> CREATE INDEX idx3 on t3(b) key_ block_ size=16384 USING HASH
GLOBAL;
Query OK, 0 rows affected
Records: 0  Duplicates: 0  Warnings: 0

gbase> SHOW CREATE TABLE t3\G
*************************** 1. row ***************************
       Table: t3
Create Table: CREATE TABLE "t3" (
  "a" int(11) DEFAULT NULL,
  "b" varchar(10) DEFAULT NULL,
```

```
   KEY "idx3" ("b") KEY_ BLOCK_ SIZE=16384 USING HASH GLOBAL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

5.1.8.5.2 DROP INDEX

Grammar format

```
DROP INDEX index_ name ON [vc_name.][database_name.]table_ name;
```

Example

Example 1: Deleting an index.

```
gbase> DROP INDEX idx3 ON t1;
Query OK, 0 rows affected
Records: 0  Duplicates: 0  Warnings: 0
```

5.1.8.6 Pre lease disk

Function Description

Pre leased disk space can be pre allocated with disk blocks in bulk, which ensures the continuity of DC data file disk blocks for columns as much as possible. When sequentially reading column DC data, performance will be significantly improved.

When creating a table, you can specify the automatic expansion size of the table. When the column files storing data in the table exceed the specified pre rented space, the system will automatically expand according to the pre rented disk size space. The pre rented disk space size can be set according to MB and GB sizes, with an expansion size within the range of [1M, 2G].

The supported operations are:

- Support the DDL operation of Alter on pre leased disks (closing pre leased disk space).
- Modify the expansion space of the pre leased disk (modify the size of the pre leased disk space).

5.1.8.6.1 Create pre leased space

Function Description

Create a table and specify the pre leased disk space size.

Grammar format

```
CREATE TABLE [IF NOT EXISTS] [vc_name.][database_name.] table_
name(col type,...)AUTOEXTEND ON NEXT NUM[M/G];
```

Table 573 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table name.
col type	Column definition.
NUM[M/G]	The size of the pre leased space, in units of M (megabytes) and G (gigabytes) for NUM, with an effective range of 1M ≤ NUM<2G.

Example

Example 1: Create a table and specify the pre leased disk space size.

```
gbase> CREATE TABLE t(nameid int, name varchar(50)) AUTOEXTEND
ON NEXT 1M;
Query OK, 0 rows affected

gbase> SHOW CREATE TABLE t\G
*************************** 1. row ***************************
       Table: t
Create Table: CREATE TABLE "t" (
  "nameid" int(11) DEFAULT NULL,
  "name" varchar(50) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace' AUTOEXTEND ON NEXT 1M
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: When the pre leased disk space size exceeds the supported range, the system prompts an error.

```
gbase>  CREATE TABLE t1(a int) AUTOEXTEND ON NEXT 3G;
ERROR 1729 (HY000): set table extend failed: must be between 1M and 2G
```

5.1.8.6.2 Modify pre leased space

Function Description

Modify the pre leased disk space size.

Grammar format

ALTER TABLE [*vc_name*.][*database_name*.] *table_ name* AUTOEXTEND ON NEXT *NUM[M/G];*

Table -574 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table name.
NUM[M/G]	Specify the size of the pre leased space, with NUM in units of M (megabytes) and G (gigabytes). The valid range is 1M ≤ NUM<2G.

Example

Example 1: Modify the specified pre leased disk space size of a table.

```
gbase> CREATE TABLE t(nameid int, name varchar(50)) AUTOEXTEND ON NEXT 1M;
Query OK, 0 rows affected

gbase> SHOW CREATE TABLE t\G
*************************** 1. row ***************************
       Table: t
Create Table: CREATE TABLE "t" (
  "nameid" int(11) DEFAULT NULL,
  "name" varchar(50) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_tablespace' AUTOEXTEND ON NEXT 1M
1 row in set (Elapsed: 00:00:00.00)

gbase> ALTER TABLE t AUTOEXTEND ON NEXT 2M;
Query OK, 0 rows affected
Records: 0  Duplicates: 0  Warnings: 0
```

```
gbase> SHOW CREATE TABLE t\G
*************************** 1. row ***************************
       Table: t
Create Table: CREATE TABLE "t" (
  "nameid" int(11) DEFAULT NULL,
  "name" varchar(50) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace' AUTOEXTEND ON NEXT 2M
1 row in set (Elapsed: 00:00:00.00)
```

5.1.8.6.3 Close pre rented space

Function Description

Close the pre leased disk space.

Grammar format

```
ALTER TABLE [vc_name.][database_name.]table_ name AUTOEXTEND OFF;
```

Table 575 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table name.

Example

Example 1: Turn off the specified pre leased disk space size of the table.

```
gbase> CREATE TABLE t(nameid int, name varchar(50)) AUTOEXTEND ON NEXT 1M;
Query OK, 0 rows affected

gbase> SHOW CREATE TABLE t\G
*************************** 1. row ***************************
       Table: t
Create Table: CREATE TABLE "t" (
  "nameid" int(11) DEFAULT NULL,
  "name" varchar(50) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_ tablespace'
AUTOEXTEND ON NEXT 1M
1 row in set (Elapsed: 00:00:00.00)
```

```
gbase> ALTER TABLE t AUTOEXTEND OFF;
Query OK, 0 rows affected
Records: 0  Duplicates: 0  Warnings: 0

gbase> SHOW CREATE TABLE t\G
*************************** 1. row ***************************
       Table: t
Create Table: CREATE TABLE "t" (
  "nameid" int(11) DEFAULT NULL,
  "name" varchar(50) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_ tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

5.1.8.7 data compression

5.1.8.7.1 overview

Data compression can reduce data storage space occupation, and the relationship between compression ratio (affecting I/O time) and decompression speed can be well controlled through appropriate parameter configuration, thereby improving query performance.

explain

- When setting the global level data compression mode, the compression parameter configuration is consistent for all data nodes within the same VC.
- The priority of data compression level is column level definition compression>table level definition compression>global definition compression method.

5.1.8.7.1.1 Data compression configuration

The unified description of compression configuration in GBase 8a MPP Cluster is as follows:

Table 576 Data Compression Configuration Description

Data compression level	Configuration method
Global compression	$ vi $GBASE_ BASE/config/gbase_ 8a_ gbase.cnf [client]

Data compression level	Configuration method
	…… [gbased] …… gbase_ compress_ method=<'*method*'> gbase_ compress_ *level=<level>* ……
Table level compression	COMPRESS(<'method'>,<level>)
Column level compression	COMPRESS(<'method'>,<level>)

Table 577 Parameter Description

Parameter Name	explain
method	Specify the compression algorithm, and when not set, show variables to display 'NO Setting'. Compression method value: ● Nozip: No compression ● HighZ: High compression ratio ● RapidZ: Fast compression ● NewRapidZ： ● STDZ： Strings in compression are not case sensitive
level	Specify a compression level of 0-9, where 1 has the lowest compression ratio and the fastest compression/decompression speed, while 9 has the opposite. When not set, show variables is displayed as 0. The default level is 0, and there are different selections for different prototype algorithms.

5.1.8.7.1.2 Legacy Data Compression Configuration

The unified description of compression configuration for versions prior to V952 in GBase 8a MPP Cluster is as follows:

Table 578 Data Compression Configuration Description

Data compression level	Configuration method
Global compression	$ vi $GBASE_ BASE/config/gbase_ 8a_ gbase.cnf [client] …… [gbased] …… gbase_ compression_ num_ method=<*num_ method*> gbase_ compression_ str_ method=<*str_ method*> ……
Table level compression	COMPRESS(<*num_method*>,<*str_method*>)
Column level compression	COMPRESS(<*num_method*>) COMPRESS(<*str_method*>)

Table 579 Parameter Description

Parameter Name	explain
num_ method	The compression method for numerical type columns in the table. The details are as follows: 0: Do not use compression 1: Use deep compression for numeric types 5: Use mild compression for numeric types
str_ method	The compression method for string type columns in the table. The details are as follows: 0: Do not use compression 3: Using deep compression for string types 5: Use mild compression for string types

5.1.8.7.1.3 Compatibility configuration correspondence

The compression algorithm of the V952 version of GBase 8a MPP CLuster is upward compatible with the original usage. The corresponding relationship is as follows:

Table 580 Corresponding Table of Compatibility between New and Old Compression Algorithms

New compression algorithm	Legacy compression algorithm
gbase_ compress_ method='NoZip' gbase_ compress_ level=0	gbase_ compression_ str_ method=0 gbase_ compression_ num_ method=0
gbase_ compress_ method='RapidZ'	gbase_ compression_ str_ method=5

New compression algorithm	Legacy compression algorithm
gbase_ compress_ level=0	gbase_ compression_ num_ method=5
gbase_ compress_ method='HighZ' gbase_ compress_ level=0	gbase_ compression_ str_ method=3 gbase_ compression_ num_ method=1
COMPRESS('NoZip',0)	COMPRESS(0,0)
COMPRESS('RapidZ',0)	COMPRESS(5,5)
COMPRESS('HighZ',0)	COMPRESS(1,3)

be careful

- When gbase_ compression_ num_ Method and GBase_ compression_ str_ Method parameter and gbase_ compress_ Method and GBase_ compress_ When both levels exist, use gbase_ compress_ Method and GBase_ compress_ The level parameter configuration shall prevail.

5.1.8.7.2 Global compression

Function Description

Set the global data compression method.

Configuration method

Modify the configuration file $GBASE for all data nodes_ BASE/config/gbase_ 8a_ gbase.cnf。

Configure the following parameters:

```
$ vi $GBASE_ BASE/config/gbase_ 8a_ gbase.cnf
[client]
……
[gbased]
……
gbase_ compress_ method=<method>
gbase_ compress_ level=<level>
……
```

be careful

- The parameter is read-only and does not support client settings. The configuration file needs to be modified.
- After modifying the configuration file parameters, the service needs to be restarted.

5.1.8.7.3 Table level compression

Function Description

Define data compression when creating or modifying tables.

explain

- When the user turns off compression in the configuration file, creating a new table does not use compression algorithms for storage. However, if a table level compression algorithm is specified when creating a new table, the data will be stored according to the specified compression algorithm.
- When users open compression in the configuration file, they use the compress (0,0) compression attribute to create or alter tables, and do not specify a compression algorithm. If no compression method is used to store data, then no compression method will be used.

5.1.8.7.3.1 Specify compression attributes when creating tables

Grammar format

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS]
[vc_name.][database_name.]table_ name
(column_definition [ , column_definition],  ... [,  key_options])
COMPRESS (<'method'>,<level>);
```

Table 581 Parameter Description

Parameter Name	explain
method	Specify the compression algorithm, and when not set, show variables to display 'NO Setting'. Compression method value: ● Nozip: No compression ● HighZ: High compression ratio ● RapidZ: Fast compression

Parameter Name	explain
	● NewRapidZ： ● STDZ： Strings in compression are not case sensitive
level	Specify a compression level of 0-9, where 1 has the lowest compression ratio and the fastest compression/decompression speed, while 9 has the opposite. When not set, show variables is displayed as 0. The default level is 0, and there are different selections for different prototype algorithms.

Example

Example 1: Define a table using compressed table syntax:

```
gbase> CREATE TABLE t1 (a int, b varchar(10)) COMPRESS('rapidz',5);
Query OK, 0 rows affected (Elapsed: 00:00:00.12)

gbase> SHOW CREATE TABLE t1\G
*************************** 1. row ***************************
       Table: t1
Create Table: CREATE TABLE "t1" (
  "a" int(11) DEFAULT NULL,
  "b" varchar(10) DEFAULT NULL
) COMPRESS('RapidZ', 5)  ENGINE=EXPRESS DEFAULT CHARSET=utf8
TABLESPACE='sys_ tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

5.1.8.7.3.2 Modifying Table Compression Properties

Grammar format

```
ALTER TABLE [IF NOT EXISTS] [vc_name.][database_name.]table_ name
ALTER COMPRESS (<'method'>,<level>);
```

Table 582 Parameter Description

Parameter Name	explain
method	Specify the compression algorithm, and when not set, show variables to display 'NO Setting'. Compression method value: ● Nozip: No compression ● HighZ: High compression ratio

Parameter Name	explain
	● RapidZ: Fast compression ● NewRapidZ： ● STDZ： Strings in compression are not case sensitive
level	Specify a compression level of 0-9, where 1 has the lowest compression ratio and the fastest compression/decompression speed, while 9 has the opposite. When not set, show variables is displayed as 0. The default level is 0, and there are different selections for different prototype algorithms.

Example

Example 1: Modify the compression type of the table compression attribute.

```
gbase> CREATE TABLE t1 (a int, b varchar(10)) COMPRESS('rapidz',5);
Query OK, 0 rows affected (Elapsed: 00:00:00.12)

gbase> SHOW CREATE TABLE t1\G
*************************** 1. row ***************************
       Table: t1
Create Table: CREATE TABLE "t1" (
  "a" int(11) DEFAULT NULL,
  "b" varchar(10) DEFAULT NULL
) COMPRESS('RapidZ', 5)  ENGINE=EXPRESS DEFAULT CHARSET=utf8
TABLESPACE='sys_ tablespace'
1 row in set (Elapsed: 00:00:00.00)

gbase> ALTER TABLE t1 ALTER COMPRESS('newrapidz',0);
Query OK, 0 rows affected (Elapsed: 00:00:00.90)

gbase> SHOW CREATE TABLE t1\G
*************************** 1. row ***************************
       Table: t1
Create Table: CREATE TABLE "t1" (
  "a" int(11) DEFAULT NULL,
  "b" varchar(10) DEFAULT NULL
) COMPRESS('NewRapidZ', 0)  ENGINE=EXPRESS DEFAULT
CHARSET=utf8 TABLESPACE='sys_ tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

5.1.8.7.4 Column level compression

Function Description

The definition of data compression for one or more columns in a table when creating or modifying it. Convenient for users to set up separately.

5.1.8.7.4.1 Create compressed columns

Grammar format

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] [[vc_name.]database_
name.] table_ name
(column_definition [ , column_definition],    ... [,    key_options])
  [table_options];

column_ definition:
column_ name data_ type [NOT NULL | NULL] [DEFAULT default_value]
COMPRESS (<'method'>,<level>)
```

Table 583 Parameter Description

Parameter Name	explain
method	Specify the compression algorithm, and when not set, show variables to display 'NO Setting'. Compression method value: ● Nozip: No compression ● HighZ: High compression ratio ● RapidZ: Fast compression ● NewRapidZ: ● STDZ: Strings in compression are not case sensitive
level	Specify a compression level of 0-9, where 1 has the lowest compression ratio and the fastest compression/decompression speed, while 9 has the opposite. When not set, show variables is displayed as 0. The default level is 0, and there are different selections for different prototype algorithms.

Example

Example 1: Define column compression for a single column.

```
gbase> CREATE TABLE t1 (a int DEFAULT NULL,b varchar(10) COMPRESS('HighZ',0));
Query OK, 0 rows affected (Elapsed: 00:00:00.22)

gbase> SHOW CREATE TABLE t1\G
*************************** 1. row ***************************
       Table: t1
Create Table: CREATE TABLE "t1" (
  "a" int(11) DEFAULT NULL,
  "b" varchar(10) DEFAULT NULL COMPRESS('HighZ', 0)
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_tablespace'
1 row in set (Elapsed: 00:00:00.01)
```

5.1.8.7.4.2 Modify compressed columns

Function Description

Modify the column compression attribute value.

Grammar format

```
ALTER TABLE [IF NOT EXISTS] [vc_name.][database_name.]table_ name
ALTER [column] column_ name COMPRESS (<'method'>,<level>);
```

Table -584 Parameter Description

Parameter Name	explain
method	Specify the compression algorithm, and when not set, show variables to display 'NO Setting'. Compression method value: ● Nozip: No compression ● HighZ: High compression ratio ● RapidZ: Fast compression ● NewRapidZ： ● STDZ： Strings in compression are not case sensitive
level	Specify a compression level of 0-9, where 1 has the lowest compression ratio and the fastest compression/decompression speed, while 9 has the opposite. When not set, show variables is displayed as 0. The default level is 0, and there are different selections for different prototype algorithms.

Example

Example 1: Modifying a non compressed column to a compressed column

```
gbase> CREATE TABLE t1 (a int DEFAULT NULL,b varchar(10) COMPRESS('HighZ',0));
Query OK, 0 rows affected (Elapsed: 00:00:00.22)

gbase> SHOW CREATE TABLE t1\G
*************************** 1. row ***************************
       Table: t1
Create Table: CREATE TABLE "t1" (
  "a" int(11) DEFAULT NULL,
  "b" varchar(10) DEFAULT NULL COMPRESS('HighZ', 0)
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_tablespace'
1 row in set (Elapsed: 00:00:00.01)

gbase> ALTER TABLE t1 ALTER a COMPRESS('rapidz',0);
Query OK, 0 rows affected (Elapsed: 00:00:00.31)

gbase> SHOW CREATE TABLE t1\G
*************************** 1. row ***************************
       Table: t1
Create Table: CREATE TABLE "t1" (
  "a" int(11) DEFAULT NULL COMPRESS('RapidZ', 0) ,
  "b" varchar(10) DEFAULT NULL COMPRESS('HighZ', 0)
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: Modifying the compression type value of a compressed column.

```
gbase> CREATE TABLE t2 (a int ,b varchar(10) NULL COMPRESS('rapidz',0));
Query OK, 0 rows affected (Elapsed: 00:00:00.13)

gbase> SHOW CREATE TABLE t2\G
*************************** 1. row ***************************
       Table: t2
Create Table: CREATE TABLE "t2" (
  "a" int(11) DEFAULT NULL,
  "b" varchar(10) DEFAULT NULL COMPRESS('RapidZ', 0)
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
```

```
tablespace'
1 row in set (Elapsed: 00:00:00.00)

gbase> ALTER TABLE t2 ALTER b COMPRESS('Newrapidz',1);
Query OK, 0 rows affected (Elapsed: 00:00:00.14)

gbase> SHOW CREATE TABLE t2\G
*************************** 1. row ***************************
       Table: t2
Create Table: CREATE TABLE "t2" (
  "a" int(11) DEFAULT NULL,
  "b" varchar(10) DEFAULT NULL COMPRESS('NewRapidZ', 1)
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

5.1.8.8 Autoincrement column

Function Description

The autoincrement column specifies auto_ The column value of the increment attribute monotonically increases (continuity is not guaranteed). Self increasing columns support use on columns of the following data types: tinyint, smallint, int, bigint. Self adding columns can uniquely identify each record in the table, making it convenient for operations such as querying, modifying, and deleting.

explain

- Each table can only have one autoincrement column, and the data types supported by autoincrement columns can only be tinyint, smallint, int, bigint;
- The self increment column of the cluster is automatically maintained by the system, and users cannot specify the starting value and step size of self increment. The self increment of the cluster monotonically increases, and continuity is not guaranteed;
- In the cluster hash distribution table, self increasing columns cannot be used as hash distribution columns;
- In the partition table, self adding columns cannot be used as partition condition columns.
- By default, DML operations are not allowed to autoincrement columns, that is:
  - Insert cannot explicitly insert data (because checking whether the inserted data and existing data are uniquely incremented in large amounts of data can seriously affect performance);
  - You can specify specific values of NULL, 0, and default for the autoincrement column insert. These three values do not affect the autoincrement column, and the autoincrement column still maintains the values maintained by the system's automatic increment;
  - Update cannot update self increasing columns;
  - Merge cannot update/insert autoincrement columns.
- To allow DML operation to autoincrement columns, it is necessary to set_ gbase_ auto_ increment_ allow_ Insert=1, at this point, the user needs to ensure that the self increasing column value is correct;

5.1.8.8.1 Self increasing DDL operation

5.1.8.8.1.1 Create self increasing columns

5.1.8.8.1.1.1CREATE TABLE

Grammar format

create [temporary] table [if not exists] [*vc_name*.][*database_name*.]*table_ name*

(*create_definition,...*)

create_ definition:

col_ name column_ definition

column_ definition:

data_ type [not null] [default default_value]

[auto_increment [primary] key]

Table -585 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table name.
data_ type	data_ The type is tinyint, smallint, int, bigint.

explain

- The attribute of the autoincrement column must be not null;
- The primary key keyword is optional (8a does not support the primary key function, where the primary key is only used for syntax compatibility);
- Each table can only have at most one autoincrement column.
- A self increasing column cannot be a hash distribution column

Example

Example 1: Creating a self increasing column on a randomly distributed table

gbase> CREATE TABLE t1(name varchar(10),id int not null primary key auto_ increment);

Query OK, 0 rows affected (Elapsed: 00:00:00.08)

gbase> SHOW CREATE TABLE t1 \G

*************************** 1. row ***************************

Table: t1

Create Table: CREATE TABLE "t1" (

"name" varchar(10) DEFAULT NULL,

```
  "id" int(11) NOT NULL AUTO_ INCREMENT,
  PRIMARY KEY ("id")
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_ tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: Creating a self increasing column on a replicated table

```
gbase> CREATE TABLE t2(a int not null primary key auto_increment, b varchar(100)) replicated;
Query OK, 0 rows affected (Elapsed: 00:00:00.08)

gbase> SHOW CREATE TABLE t2 \G
*************************** 1. row ***************************
       Table: t2
Create Table: CREATE TABLE "t2" (
  "a" int(11) NOT NULL AUTO_ INCREMENT,
  "b" varchar(100) DEFAULT NULL,
  PRIMARY KEY ("a")
) ENGINE=EXPRESS REPLICATED DEFAULT CHARSET=utf8 TABLESPACE='sys_ tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 3: Creating a self increasing column on a hash distribution table

```
gbase> CREATE TABLE t3(name varchar(10),id int not null primary key auto_ increment) distributed by ('name');
Query OK, 0 rows affected (Elapsed: 00:00:00.08)

gbase> SHOW CREATE TABLE t3 \G
*************************** 1. row ***************************
```

```
           Table: t3
Create Table: CREATE TABLE "t3" (
   "name" varchar(10) DEFAULT NULL,
   "id" int(11) NOT NULL AUTO_ INCREMENT,
   PRIMARY KEY ("id")
) ENGINE=EXPRESS DISTRIBUTED BY('name') DEFAULT CHARSET=utf8
TABLESPACE='sys_ tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 4: Creating self increasing columns on a partition table

```
gbase> CREATE TABLE t4(i int default 0,id int auto_increment primary
key )
 partition by range(i) partitions 2
 subpartition by hash(i) subpartitions 2
 (
     partition    p0 values less than (50001)
      (subpartition sb0,
        subpartition sb1
      ),
     partition    p1 values less than (100001)
      (subpartition sb10,
        subpartition sb11
      )   ) ;
Query OK, 0 rows affected (Elapsed: 00:00:00.11)

gbase> SHOW CREATE TABLE t4 \G
*************************** 1. row ***************************
           Table: t4
```

```
Create Table: CREATE TABLE "t4" (
   "i" int(11) DEFAULT '0',
   "id" int(11) NOT NULL AUTO_ INCREMENT,
   PRIMARY KEY ("id")
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
 PARTITION BY RANGE (i)
SUBPARTITION BY HASH (i)
(PARTITION p0 VALUES LESS THAN (50001)
 (SUBPARTITION sb0 ENGINE = EXPRESS,
   SUBPARTITION sb1 ENGINE = EXPRESS),
 PARTITION p1 VALUES LESS THAN (100001)
 (SUBPARTITION sb10 ENGINE = EXPRESS,
   SUBPARTITION sb11 ENGINE = EXPRESS))
1 row in set (Elapsed: 00:00:00.01)
```

5.1.8.8.1.1.2CREATE TABLE LIKE

Function Description

The source table contains self increasing columns, and the target table of create table like inherits the self increasing column attribute.

Example

```
gbase> CREATE TABLE t1_ 1 like t1;
Query OK, 0 rows affected (Elapsed: 00:00:00.07)

gbase> SHOW CREATE TABLE t1_ 1 \G
*************************** 1. row ***************************
          Table: t1_ one
```

```
Create Table: CREATE TABLE "t1_1" (
   "name" varchar(10) DEFAULT NULL,
   "id" int(11) NOT NULL AUTO_ INCREMENT,
   PRIMARY KEY ("id")
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

5.1.8.8.1.1.3CREATE TABLE AS SELECT

Create table as select cannot automatically inherit autoincrement attributes from the source table. The related attributes of the column must be specified in the create statement to create the table. Autoincrement column attributes can be inherited in the following two situations:

- When creating a table, define a self increasing column, and the source table does not specify the corresponding column for the self increasing column.

  create table td(a int auto_increment primary key,b int ,c int) as select xxx as b from ts;

- When creating a table, define autoincrement columns. The corresponding columns in the source table are also autoincrement columns, and the data type range of the autoincrement columns in the source table is less than or equal to the range of the autoincrement columns in the table to be built.

  create table td(a int auto_increment primary key,b int ,c int) as SELECT xxx as a, yyy as b from ts;

  The xxx column in the ts table is a self increasing column.

Example

Example 1: The source table does not specify the column corresponding to the autoincrement column

```
gbase>  CREATE TABLE ts (a1 int ,b1 int ,c1 int) ;
Query OK, 0 rows affected (Elapsed: 00:00:00.01)
```

```
gbase>    INSERT INTO ts values(1,2,3),(7,8,9);
Query OK, 2 rows affected (Elapsed: 00:00:00.01)
Records: 2    Duplicates: 0    Warnings: 0 Total: 2

gbase>    CREATE TABLE td(a int auto_increment primary key,b int ,c int)
as SELECT b1 as b FROM ts;
Query OK, 2 rows affected (Elapsed: 00:00:00.02)
Records: 2    Duplicates: 0    Warnings: 0

gbase> SHOW CREATE TABLE td \G
*************************** 1. row ***************************
            Table: td
Create Table: CREATE TABLE "td" (
   "a" int(11) NOT NULL AUTO_ INCREMENT,
   "c" int(11) DEFAULT NULL,
   "b" int(11) DEFAULT NULL,
   PRIMARY KEY ("a")
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: The column corresponding to the autoincrement column is specified in the source table, and the corresponding column in the source table is an autoincrement column.

```
gbase>    CREATE TABLE ts (a1 int auto_increment primary key,b1 int ,c1
int) ;
Query OK, 0 rows affected (Elapsed: 00:00:00.10)

gbase> INSERT INTO ts(b1,c1) values(2,3),(8,9);
Query OK, 2 rows affected (Elapsed: 00:00:00.11)
```

```
Records: 2  Duplicates: 0  Warnings: 0


gbase>  CREATE TABLE td(a int auto_increment primary key,b int ,c int)
as SELECT a1 as a, b1 as b FROM ts;

Query OK, 2 rows affected (Elapsed: 00:00:00.25)


gbase> SHOW CREATE TABLE td \G

*************************** 1. row ***************************
       Table: td
Create Table: CREATE TABLE "td" (
  "c" int(11) DEFAULT NULL,
  "a" int(11) NOT NULL AUTO_ INCREMENT,
  "b" int(11) DEFAULT NULL,
  PRIMARY KEY ("a")
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)

gbase> SELECT * FROM td;

+------+---+------+

| c    | a | b    |

+------+---+------+

| NULL | 1 |    2 |

| NULL | 3 |    8 |

+------+---+------+

2 rows in set (Elapsed: 00:00:00.04)
```

5.1.8.8.1.2 Add/Remove Self Increasing Columns

Function Description

Create an incremental column through the Alter Table. When adding a self increasing column to a non empty table, the added self increasing column will be automatically filled with data.

Grammar format

```
alter table [vc_name.][database_name.]table_ name alter_ specification
[,alter_specification] ...

alter_ specification:

        |add [column] col_ name column_ definition

        | drop [column] col_ name
```

Example

Example 1: Adding a self increasing column to an empty table

```
gbase>    CREATE TABLE t1 (a VARCHAR(10));

Query OK, 0 rows affected (Elapsed: 00:00:00.07)



gbase>    ALTER TABLE t1 ADD COLUMN id INT NOT NULL AUTO_
INCREMENT PRIMARY KEY;

Query OK, 0 rows affected (Elapsed: 00:00:00.10)

Records: 0    Duplicates: 0    Warnings: 0



gbase> SHOW CREATE TABLE t1\G

*************************** 1. row ***************************

            Table: t1

Create Table: CREATE TABLE "t1" (

   "a" varchar(10) DEFAULT NULL,

   "id" int(11) NOT NULL AUTO_ INCREMENT,

   PRIMARY KEY ("id")

)  ENGINE=EXPRESS  DEFAULT  CHARSET=utf8  TABLESPACE='sys_
tablespace'

1 row in set (Elapsed: 00:00:00.00)
```

Example 2: Adding a self increasing column to a non empty table

```
gbase>  CREATE TABLE t1 (a VARCHAR(10));
Query OK, 0 rows affected (Elapsed: 00:00:00.07)

gbase>  INSERT INTO t1 VALUES('a'),('b'),('c');
Query OK, 3 rows affected (Elapsed: 00:00:00.07)
Records: 3  Duplicates: 0  Warnings: 0

gbase>  ALTER TABLE t1 ADD COLUMN id INT NOT NULL AUTO_
INCREMENT PRIMARY KEY;
Query OK, 3 rows affected (Elapsed: 00:00:00.10)
Records: 3  Duplicates: 3  Warnings: 0

base>  SHOW CREATE TABLE t1\G
*************************** 1. row ***************************
       Table: t1
Create Table: CREATE TABLE "t1" (
  "a" varchar(10) DEFAULT NULL,
  "id" int(11) NOT NULL AUTO_ INCREMENT,
  PRIMARY KEY ("id")
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)

gbase> SELECT * FROM t1;
+------+----+
| a    | id |
+------+----+
```

```
| a      |    2 |

| b      |    6 |

| c      |    10 |

+------+----+

3 rows in set (Elapsed: 00:00:00.02)
```

Example 3: Deleting a self increasing column

```
gbase> CREATE TABLE t1 (a1 INT AUTO_INCREMENT PRIMARY KEY,b1 INT ,c1 INT) ;

Query OK, 0 rows affected (Elapsed: 00:00:00.10)



gbase>    INSERT INTO t1(b1,c1) VALUES(2,3),(8,9);

Query OK, 2 rows affected (Elapsed: 00:00:00.09)

Records: 2    Duplicates: 0    Warnings: 0



gbase>    SELECT * FROM t1;

+----+------+------+

| a1 | b1     | c1     |

+----+------+------+

|   2 |       2 |       3 |

|   6 |       8 |       9 |

+----+------+------+

2 rows in set (Elapsed: 00:00:00.03)



gbase>    SHOW CREATE TABLE t1 \G

*************************** 1. row ***************************

            Table: t1

Create Table: CREATE TABLE "t1" (
```

```
  "a1" int(11) NOT NULL AUTO_ INCREMENT,
  "b1" int(11) DEFAULT NULL,
  "c1" int(11) DEFAULT NULL,
  PRIMARY KEY ("a1")
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)

gbase> ALTER TABLE t1 DROP COLUMN a1;
Query OK, 2 rows affected (Elapsed: 00:00:00.10)
Records: 2 Duplicates: 2 Warnings: 0

gbase> SHOW CREATE TABLE t1 \G
*************************** 1. row ***************************
       Table: t1
Create Table: CREATE TABLE "t1" (
  "b1" int(11) DEFAULT NULL,
  "c1" int(11) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.01)

gbase> SELECT * FROM t1;
+------+------+
| b1   | c1   |
+------+------+
|    2 |    3 |
```

```
|      8 |      9 |
+------+------+
2 rows in set (Elapsed: 00:00:00.03)
```

5.1.8.8.2 Self increasing DML operation

Tables containing self increasing columns are not allowed to directly operate on self increasing columns during DML operations. For example, when inserting, the autoincrement column is not allowed to specify a value, and when inserting select, the autoincrement column is not allowed to be specified, but when setting parameters_ gbase_ auto_ increment_ allow_ When insert=1, DML operation is allowed to autoincrement columns, but the user needs to ensure that the autoincrement column value is correct.

5.1.8.8.2.1INSERT VALUES

Function Description

When executing Insert Values, the autoincrement column can either not specify a value or specify specific values such as NULL, 0, or default, which will be automatically maintained according to the cluster autoincrement rules.

Example

Example 1: Create a self increasing column on a randomly distributed table, and automatically maintain the data of the self increasing column.

```
create table t1(a int auto_increment primary key,b varchar(100), c int);
insert into t1(b,c) values('a1',1),('a2',2),('a3',3),('a4',4),('a5',5);
gbase> SELECT * from t1;
+---+------+------+
| a | b    | c    |
+---+------+------+
| 2 | a1   |    1 |
| 6 | a2   |    2 |
| 10 | a3  |    3 |
| 14 | a4  |    4 |
```

```
| 18 | a5     |       5 |
+---+------+------+
5 rows in set (Elapsed: 00:00:00.03)
```

Example 2: Creating a self increasing column on a hash distribution table

```
create table t2(a int auto_increment primary key,b varchar(100), c int)
distributed by ('b');
insert into t2(b,c) values('a1',1),('a2',2),('a3',3),('a4',4),('a5',5);
gbase> SELECT * FROM t2;
+---+------+------+
| a | b       | c       |
+---+------+------+
| 2 | a1     |       1 |
| 6 | a5     |       5 |
| 1 | a3     |       3 |
| 3 | a2     |       2 |
| 4 | a4      |        4 |
+---+------+------+
5 rows in set (Elapsed: 00:00:00.02)

gbase> SELECT * FROM t2 ORDER BY a;
+---+------+------+
| a | b       | c       |
+---+------+------+
| 1 | a3     |       3 |
| 2 | a1     |       1 |
| 3 | a2     |       2 |
| 4 | a4     |       4 |
```

```
| 6 | a5    |      5 |
+---+------+------+
5 rows in set (Elapsed: 00:00:00.04)
```

Example 3: Insert 0, null, default into a self increasing column.

```
CREATE TABLE t3(a INT AUTO_INCREMENT PRIMARY KEY,b VARCHAR(100), c INT);
gbase> INSERT INTO t3 VALUES(0,'a1',1);
Query OK, 1 row affected (Elapsed: 00:00:00.05)
gbase> SELECT * FROM t3;
+---+------+------+
| a | b    | c    |
+---+------+------+
| 2 | a1   |    1 |
+---+------+------+
1 row in set (Elapsed: 00:00:00.01)

gbase> INSERT INTO t3 VALUES(NULL,'a2',2);
Query OK, 1 row affected (Elapsed: 00:00:00.04)

gbase> SELECT * FROM t3;
+---+------+------+
| a | b    | c    |
+---+------+------+
| 2 | a1   |    1 |
| 4 | a2   |    2 |
+---+------+------+
```

```
2 rows in set (Elapsed: 00:00:00.00)

gbase> INSERT INTO t3 VALUES(NULL,'a3',3);
Query OK, 1 row affected (Elapsed: 00:00:00.05)

gbase> SELECT * FROM t3;
+---+------+------+
| a | b    | c    |
+---+------+------+
| 2 | a1   |    1 |
| 4 | a2   |    2 |
| 6 | a3   |    3 |
+---+------+------+
3 rows in set (Elapsed: 00:00:00.01)

gbase> INSERT INTO t3 VALUES(DEFAULT,'a4',4);
Query OK, 1 row affected (Elapsed: 00:00:00.04)

gbase> SELECT * FROM t3;
+---+------+------+
| a | b    | c    |
+---+------+------+
| 2 | a1   |    1 |
| 4 | a2   |    2 |
| 6 | a3   |    3 |
| 8 | a4   |    4 |
+---+------+------+
```

```
4 rows in set (Elapsed: 00:00:00.01)
```

5.1.8.8.2.2INSERT SELECT

Function Description

When executing Insert INTO... SELECT..., the autoincrement column does not allow specifying values, but 0 and NULL can be specified.

Example

Example 1: The autoincrement column does not specify a value and is automatically maintained.

```
CREATE TABLE t1(a INT AUTO_INCREMENT PRIMARY KEY,b
VARCHAR(100), c INT);

INSERT INTO t1(b,c) VALUES('a1',1),('a2',2),('a3',3),('a4',4),('a5',5);

CREATE TABLE t2(a INT AUTO_INCREMENT PRIMARY KEY,b
VARCHAR(100), c INT) DISTRIBUTED BY ('b');


gbase> select * from t1;

+----+------+------+

| a  | b    | c    |

+----+------+------+

|  2 | a1   |    1 |

|  6 | a2   |    2 |

| 10 | a3   |    3 |

| 14 | a4   |    4 |

| 18 | a5   |    5 |

+----+------+------+

5 rows in set (Elapsed: 00:00:00.03)


gbase> INSERT INTO t2(b,c) SELECT b,c FROM t1;
```

```
Query OK, 5 rows affected (Elapsed: 00:00:00.15)

Records: 5  Duplicates: 0  Warnings: 0


gbase> SELECT * FROM t2;

+---+------+------+

| a | b    | c    |

+---+------+------+

| 3 | a2   |    2 |

| 2 | a1   |    1 |

| 6 | a5   |    5 |

| 4 | a4   |    4 |

| 1 | a3   |    3 |

+---+------+------+

5 rows in set (Elapsed: 00:00:00.02)
```

Example 2: Insert 0, null, and automatically maintain a self increasing column.

```
CREATE TABLE t1(a INT AUTO_INCREMENT PRIMARY KEY,b VARCHAR(100), c INT);

INSERT INTO t1(b,c) VALUES('a1',1),('a2',2),('a3',3),('a4',4),('a5',5);

CREATE TABLE t2(a INT AUTO_INCREMENT PRIMARY KEY,b VARCHAR(100), c INT) DISTRIBUTED BY ('b');

gbase> INSERT INTO t2 SELECT 0,b,c FROM t1;

Query OK, 5 rows affected (Elapsed: 00:00:00.11)

Records: 5  Duplicates: 0  Warnings: 0


gbase> SELECT * FROM t2;

+---+------+------+

| a | b    | c    |
```

```
+---+------+------+
| 3 | a2   |    2 |
| 2 | a1   |    1 |
| 6 | a5   |    5 |
| 4 | a4   |    4 |
| 1 | a3   |    3 |
+---+------+------+
5 rows in set (Elapsed: 00:00:00.02)

gbase> INSERT INTO t2 SELECT NULL,b,c FROM t1;
Query OK, 5 rows affected (Elapsed: 00:00:00.11)
Records: 5  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t2 ORDER BY a;
+----+------+------+
| a  | b    | c    |
+----+------+------+
|  1 | a3   |    3 |
|  2 | a1   |    1 |
|  3 | a2   |    2 |
|  4 | a4   |    4 |
|  6 | a5   |    5 |
|  7 | a3   |    3 |
|  8 | a1   |    1 |
|  9 | a2   |    2 |
| 10 | a4   |    4 |
| 12 | a5   |    5 |
```

```
+----+------+------+

10 rows in set (Elapsed: 00:00:00.08)
```

5.1.8.8.2.3DELETE

Function Description

When the Delete statement deletes data, the value of the self increasing column will not be reclaimed, and the value will continue to monotonically increase

Example

```
create table t1(a int auto_increment primary key,b varchar(100), c int);

insert into t1(b,c) values('a1',1),('a2',2),('a3',3),('a4',4),('a5',5);

gbase> SELECT * FROM t1;

+----+------+------+

| a    | b       | c       |

+----+------+------+

|    2 | a1     |       1 |

|    6 | a2     |       2 |

| 10 | a3     |       3 |

| 14 | a4     |       4 |

| 18 | a5     |       5 |

+----+------+------+

5 rows in set (Elapsed: 00:00:00.02)



gbase> delete from t1 where a = 3;

Query OK, 1 row affected (Elapsed: 00:00:00.06)



gbase> SELECT * FROM t1;

+----+------+------+
```

```
| a    | b       | c       |
+----+------+------+
|    2 | a1      |       1 |
| 10 | a3      |       3 |
| 14 | a4      |       4 |
| 18 | a5      |       5 |
+----+------+------+
4 rows in set (Elapsed: 00:00:00.02)

gbase> insert into t1(b,c) values('a13',13);
Query OK, 1 row affected (Elapsed: 00:00:00.07)

gbase> SELECT * FROM t1;
+----+------+------+
| a    | b       | c       |
+----+------+------+
|    2 | a1      |       1 |
| 10 | a3      |       3 |
| 14 | a4      |       4 |
| 18 | a5      |       5 |
| 20 | a13    |      13 |
+----+------+------+
5 rows in set (Elapsed: 00:00:00.02)
```

5.1.8.8.2.4UPDATE

Function Description

When updating, self increasing columns cannot be updated.

Example

```
create table t2(a int auto_increment primary key,b varchar(100), c int) distributed by ('b');

insert into t2(b,c) values('a1',1),('a2',2),('a3',3),('a4',4),('a5',5);


gbase> SELECT * FROM t2;

+---+------+------+

| a | b    | c    |

+---+------+------+

| 3 | a2   |    2 |

| 1 | a3   |    3 |

| 2 | a1   |    1 |

| 6 | a5   |    5 |

| 4 | a4   |    4 |

+---+------+------+

5 rows in set (Elapsed: 00:00:00.02)


gbase> update t2 set c = 144 where b = 'a2';

Query OK, 1 row affected (Elapsed: 00:00:00.08)

Rows matched: 1  Changed: 1  Warnings: 0


gbase> update t2 set a = 441 where b = 'a2';

ERROR 1235 (42000): This version of GBase doesn't yet support 'update/merge on auto_ increment column'

gbase> SELECT * from t2;

+---+------+------+

| a | b    | c    |
```

```
+---+------+------+

| 1 | a3   |    3 |

| 3 | a2   |  144 |

| 2 | a1   |    1 |

| 6 | a5   |    5 |

| 4 | a4   |    4 |

+---+------+------+

5 rows in set (Elapsed: 00:00:00.02)
```

5.1.8.8.2.5MERGE

Function Description

When merging, the update section cannot specify an update autoincrement column, and the insert section cannot specify an autoincrement column.

Example

```
CREATE TABLE t1(a INT AUTO_INCREMENT PRIMARY KEY,b VARCHAR(100), c INT) DISTRIBUTED BY ('b');

CREATE TABLE t2(a INT AUTO_INCREMENT PRIMARY KEY,b VARCHAR(100), c INT) DISTRIBUTED BY ('b');

INSERT INTO t1(b,c) VALUES('a',1),('b',2);

INSERT INTO t2(b,c) VALUES('a',7),('b',8),('c',3),('d',4);


gbase> SELECT * FROM t1;

+---+------+------+

| a | b    | c    |

+---+------+------+

| 2 | b    |    2 |

| 3 | a    |    1 |

+---+------+------+
```

```
2 rows in set (Elapsed: 00:00:00.01)


gbase> SELECT * FROM t2;
+---+------+------+
| a | b    | c    |
+---+------+------+
| 4 | c    |    3 |
| 2 | b    |    8 |
| 6 | d    |    4 |
| 3 | a    |    7 |
+---+------+------+
4 rows in set (Elapsed: 00:00:00.01)
gbase> MERGE INTO t1 USING t2 ON t1.b = t2.b WHEN MATCHED THEN UPDATE SET t1.a = t2.c WHEN NOT MATCHED THEN INSERT(t1.b,t1.c) VALUES(t2.b,t2.c);
ERROR 1235 (42000): This version of GBase doesn't yet support 'update/merge on auto_ increment column'
gbase> MERGE INTO t1 USING t2 ON t1.b = t2.b WHEN MATCHED THEN UPDATE SET t1.c = t2.c WHEN NOT MATCHED THEN INSERT(t1.a,t1.c) VALUES(t2.a,t2.c);
ERROR 1235 (42000): This version of GBase doesn't yet support 'update/merge on auto_ increment column'


gbase> MERGE INTO t1 USING t2 ON t1.b = t2.b WHEN MATCHED THEN UPDATE SET t1.c = t2.c WHEN NOT MATCHED THEN INSERT(t1.b,t1.c) VALUES(t2.b,t2.c);
Query OK, 4 rows affected (Elapsed: 00:00:00.05)
Rows matched: 4  Changed: 4  Warnings: 0
```

```
gbase> SELECT * FROM t1;
+---+------+------+
| a | b    | c    |
+---+------+------+
| 3 | a    |    7 |
| 7 | c    |    3 |
| 2 | b    |    8 |
| 5 | d    |    4 |
+---+------+------+
4 rows in set (Elapsed: 00:00:00.01)
```

5.1.8.8.2.6LOADER

Function Description

To load data into a table containing self increasing columns, it is necessary to use table fields to specify the column method for data loading, and it is not allowed to specify self increasing columns.

Example

```
CREATE TABLE lineitem
(
a int auto_ increment primary key,
L_ ORDERKEY       INT NOT NULL,
L_ PARTKEY        INTEGER NOT NULL,
L_ SUPPKEY        INTEGER NOT NULL,
L_ LINENUMBER    INTEGER NOT NULL,
L_ QUANTITY       DECIMAL(15,2) NOT NULL,
L_ EXTENDEDPRICE    DECIMAL(15,2) NOT NULL,
L_ DISCOUNT       DECIMAL(15,2) NOT NULL,
```

```
L_ TAX              DECIMAL(15,2) NOT NULL,
L_ RETURNFLAG    CHAR(1) NOT NULL,
L_ LINESTATUS    CHAR(1) NOT NULL,
L_ SHIPDATE       DATE NOT NULL,
L_ COMMITDATE    DATE NOT NULL,
L_ RECEIPTDATE DATE NOT NULL,
L_ SHIPINSTRUCT CHAR(25) NOT NULL,
L_ SHIPMODE         CHAR(10) NOT NULL,
L_ COMMENT          VARCHAR(44) NOT NULL
);

gbase> LOAD DATA INFILE ' http://192.168.154.99/tpch1s/lineitem.tbl ' INTO TABLE test.lineitem    FIELDS terminated by '|' table_ fields 'L_ ORDERKEY,L_    PARTKEY,L_    SUPPKEY,L_    LINENUMBER,L_ QUANTITY,L_    EXTENDEDPRICE,L_    DISCOUNT,L_    TAX,L_ RETURNFLAG,L_ LINESTATUS,L_ SHIPDATE,L_ COMMITDATE,L_ RECEIPTDATE,L_ SHIPINSTRUCT,L_ SHIPMODE,L_ COMMENT';
Query OK, 6001215 rows affected (Elapsed: 00:00:07.60)
Task 5250 finished, Loaded 6001215 records, Skipped 0 records

gbase> SELECT count(distinct(a)) from lineitem;
+--------------------+
| count(distinct(a)) |
+--------------------+
|                6001215 |
+--------------------+
1 row in set (Elapsed: 00:00:01.09)
```

5.1.8.8.2.7LAST_ INSERT_ ID()

Function Description

After inserting, call this function to obtain the self increasing column value just inserted. When inserting multiple rows at a time, this function only returns the value generated by inserting the first row.

Example

```
gbase> CREATE TABLE t1(a INT AUTO_INCREMENT PRIMARY KEY,b
VARCHAR(100));
Query OK, 0 rows affected (Elapsed: 00:00:00.11)

gbase> insert into t1(b) values('b1');
Query OK, 1 row affected (Elapsed: 00:00:00.09)

gbase> insert into t1(b) values('b2');
Query OK, 1 row affected (Elapsed: 00:00:00.09)

gbase> select * from t1;
+---+------+
| a | b    |
+---+------+
| 2 | b1   |
| 4 | b2   |
+---+------+
2 rows in set (Elapsed: 00:00:00.02)

gbase> SELECT last_ insert_ id() from t1;
+------------------+
```

```
| last_ insert_ id() |
+------------------+
|                  4 |
|                  4 |
+------------------+
2 rows in set (Elapsed: 00:00:00.02)

gbase>    insert into t1(b) values('b3'),('b4');
Query OK, 2 rows affected (Elapsed: 00:00:00.05)
Records: 2    Duplicates: 0    Warnings: 0

gbase> select * from t1;
+----+------+
| a    | b       |
+----+------+
|    2 | b1      |
|    4 | b2      |
|    6 | b3      |
| 10 | b4      |
+----+------+
4 rows in set (Elapsed: 00:00:00.04)

gbase> SELECT last_ insert_ id() from t1;
+------------------+
| last_ insert_ id() |
+------------------+
|                  6 |
```

```
|                    6 |
|                    6 |
|                    6 |
+------------------+
4 rows in set (Elapsed: 00:00:00.02)

gbase> create table t2(a int auto_increment primary key,b varchar(100));
Query OK, 0 rows affected (Elapsed: 00:00:00.13)

gbase> insert into t2(b) values('a9');
Query OK, 1 row affected (Elapsed: 00:00:00.09)

gbase> SELECT last_ insert_ id() from t2;
+------------------+
| last_ insert_ id() |
+------------------+
|                    2 |
+------------------+
1 row in set (Elapsed: 00:00:00.03)
```

5.1.8.9 Mixed storage of rows and columns

5.1.8.9.1 Definition of mixed storage of rows and columns

Row column mixed storage refers to the mixed storage of row and column storage, which combines the data of certain columns on the basis of existing column storage and stores them as one column. When there are many columns and the accessed data records are very discrete, redundant row storage can effectively improve I/O performance.

Function Description

Mixed row and column storage has the following functions:

- Support SQL syntax, including defining column column mixed storage during table creation, creating column column mixed storage for existing tables, and deleting column column mixed storage;
- Support rapid creation and parallel creation of mixed storage of rows and columns;
- Mixed storage of rows and columns supports compressed storage;
- Improve I/O performance, allowing column column mixed storage to read data in smaller granularity Data Pages instead of DCs;
- The system will automatically determine whether a scenario requires the use of mixed row and column data;
- Flexible storage redundancy methods, allowing users to customize data storage and redundancy methods;
- Row and column mixed storage maintenance, DML statements automatically maintain row and column mixed storage, including Insert, Quick UPDATE, DELETE, Load, etc.

Use constraints

- Cannot have the same name as other columns in the table (including mixed rows and columns).
- The same field is not allowed to appear in two column column mixed storage definitions.
- Except for deleting column column mixed storage statements, column column mixed storage is not allowed to be directly referenced in any statement.
- The definition of mixed row and column storage cannot be modified. If modification is indeed necessary, it can only be deleted first and then created based on the new definition.
- The physical columns included in the definition of mixed column and column storage do not allow deletion or modification of data types, but column names and order in the table can be modified.
- Only 0, 3, and 5 compression methods are allowed for row and column storage. Using other compression methods may result in errors.
- The name of a row or column cannot be the same as the index name in the table.

5.1.8.9.2 Specify rows and columns when creating a table

Grammar format

```
CREATE TABLE [[vc_name.]database_ name.] table_ name (column-definitions,
        [GROUPED_DEFINITIONS]
);
GROUPED_ DEFINITION:
```

```
GROUPED
[grouped_name](column_references)[COMPRESS(<'method'>,<level>)]
```

Table -586 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table Name
GROUPED	Keyword, indicating that a row or column is defined.
grouped_ name	Represents the name of a row or column, which can be specified. If no name is specified, it defaults to the following column_ The name of the first column in references. If the name is duplicated, add "_ #" after the name (# is a number starting from 2).
column_ references	A collection of physical columns contained in a row or column, separated by ','.
COMPRESS(<'method'>,<level>)	Specify a compression algorithm for rows and columns. Please refer to the compression section for specific usage.

Example

Example 1: When creating a table, create a row storage column and specify its name. The row storage column uses a compression algorithm.

```
gbase> CREATE TABLE t(a int,b int,c int,d int,GROUPED ga(b,c) COMPRESS('rapidz',0));
Query OK, 0 rows affected (Elapsed: 00:00:00.19)

gbase> show create table t\G
*************************** 1. row ***************************
       Table: t
Create Table: CREATE TABLE "t" (
  "a" int(11) DEFAULT NULL,
  "b" int(11) DEFAULT NULL,
  "c" int(11) DEFAULT NULL,
  "d" int(11) DEFAULT NULL,
  GROUPED "ga" ("b","c") COMPRESS('RapidZ', 0)
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
```

```
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: When creating a table, create a row or column without specifying the name of the row or column.

```
gbase> CREATE TABLE t(a int,b int,c int,d int,GROUPED (b,c),GROUPED
(d));
Query OK, 0 rows affected

gbase> SHOW CREATE TABLE t\G
*************************** 1. row ***************************
           Table: t
Create Table: CREATE TABLE "t" (
  "a" int(11) DEFAULT NULL,
  "b" int(11) DEFAULT NULL,
  "c" int(11) DEFAULT NULL,
  "d" int(11) DEFAULT NULL,
  GROUPED "b" ("b","c"),
  GROUPED "d" ("d")
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 3: When creating a table, create rows and columns, and specify the names of some rows and columns. If the names of rows and columns are duplicate, add "_ #" after the name (# is a number starting from 2).

```
gbase> DROP TABLE IF EXISTS t;
Query OK, 0 rows affected

gbase> CREATE TABLE t(a int,b int,c int,d int,GROUPED
a(b,c),GROUPED (a));
Query OK, 0 rows affected

gbase> SHOW CREATE TABLE t\G
*************************** 1. row ***************************
           Table: t
Create Table: CREATE TABLE "t" (
  "a" int(11) DEFAULT NULL,
  "b" int(11) DEFAULT NULL,
  "c" int(11) DEFAULT NULL,
  "d" int(11) DEFAULT NULL,
  GROUPED "a" ("b","c"),
```

```
   GROUPED "a_2" ("a")
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 4: When creating a table, creating multiple rows and columns, repeatedly specifying row and column names, returns error code 1061.

```
gbase> DROP TABLE IF EXISTS t;
Query OK, 0 rows affected

gbase> CREATE TABLE t (a int,b int,c int,d int, GROUPED a(b,c),
GROUPED a(d));
ERROR 1702 (HY000): gcluster table error: Duplicate key name 'a'.
```

Example 5: When creating a table, create multiple rows and columns, and specify the names of the rows and columns, using compressed table syntax.

```
gbase>   DROP TABLE IF EXISTS t;
Query OK, 0 rows affected

gbase> CREATE TABLE t(a int,b int,c int,d int,GROUPED
a(b,c),GROUPED b(a,d)) COMPRESS('RapidZ', 0);
Query OK, 0 rows affected

gbase> SHOW CREATE TABLE t \G
*************************** 1. row ***************************
       Table: t
Create Table: CREATE TABLE "t" (
  "a" int(11) DEFAULT NULL,
  "b" int(11) DEFAULT NULL,
  "c" int(11) DEFAULT NULL,
  "d" int(11) DEFAULT NULL,
  GROUPED "a" ("b","c"),
  GROUPED "b" ("a","d")
) COMPRESS('RapidZ', 0)   ENGINE=EXPRESS DEFAULT CHARSET=utf8
TABLESPACE='sys_ tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 6: When creating a table, create multiple rows and columns, and specify the names of the rows and columns. Both the rows and columns and the table use compression syntax.

```
gbase> DROP TABLE IF EXISTS t;
```

```
Query OK, 0 rows affected

gbase> CREATE TABLE t(a int,b int,c int,d int,GROUPED a(b,c) ,GROUPED b(a,d) COMPRESS('HighZ', 0)) COMPRESS('RapidZ', 0);
Query OK, 0 rows affected

gbase> SHOW CREATE TABLE t \G
*************************** 1. row ***************************
       Table: t
Create Table: CREATE TABLE "t" (
  "a" int(11) DEFAULT NULL,
  "b" int(11) DEFAULT NULL,
  "c" int(11) DEFAULT NULL,
  "d" int(11) DEFAULT NULL,
  GROUPED "a" ("b","c"),
  GROUPED "b" ("a","d") COMPRESS('HighZ', 0)
) COMPRESS('RapidZ', 0) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_ tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

5.1.8.9.3 Create rows and columns when modifying tables

Grammar format

```
ALTER TABLE [[vc_name.]database_ name.] table_ name ADD GROUPED_ DEFINITION ;
GROUPED_ DEFINITION:
    GROUPED [grouped_name](column_references)[COMPRESS(<'method'>,<level>)]
```

Table 587 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table Name
GROUPED	Keyword, indicating that a row or column is defined.
grouped_ name	Represents the name of a row or column, which can be specified. If no name is specified, it defaults to the following column_ The name of the first column in

Parameter Name	explain
	references. If the name is duplicated, add "_ #" after the name (# is a number starting from 2).
column_ references	A collection of physical columns contained in a row or column, separated by ','.
COMPRESS(<'method'>,<level>)	Specify a compression algorithm for rows and columns. Please refer to the compression section for specific usage.

Example

Example 1: When creating a table, do not create a row or column, and use the Alter Table statement to create a row or column.

```
gbase> DROP TABLE t2;
Query OK, 0 rows affected

gbase> CREATE TABLE t2(a int,b int,c int,d int);
Query OK, 0 rows affected

gbase> ALTER TABLE t2 ADD GROUPED (a,b,c);
Query OK, 0 rows affected
Records: 0  Duplicates: 0  Warnings: 0

gbase> SHOW CREATE TABLE t2 \G
*************************** 1. row ***************************
       Table: t2
Create Table: CREATE TABLE "t2" (
  "a" int(11) DEFAULT NULL,
  "b" int(11) DEFAULT NULL,
  "c" int(11) DEFAULT NULL,
  "d" int(11) DEFAULT NULL,
  GROUPED "a" ("a","b","c")
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: When creating a table, create some columns as row storage columns, and use the Alter Table statement to add row storage columns.

```
gbase> DROP TABLE t2;
Query OK, 0 rows affected
```

```
gbase> CREATE TABLE t2(a int,b int,c int,d int, GROUPED a(a,b));
Query OK, 0 rows affected

gbase> ALTER TABLE t2 ADD GROUPED c(c,d);
Query OK, 0 rows affected
Records: 0    Duplicates: 0    Warnings: 0

gbase> SHOW CREATE TABLE t2 \G
*************************** 1. row ***************************
           Table: t2
Create Table: CREATE TABLE "t2" (
   "a" int(11) DEFAULT NULL,
   "b" int(11) DEFAULT NULL,
   "c" int(11) DEFAULT NULL,
   "d" int(11) DEFAULT NULL,
   GROUPED "a" ("a","b"),
   GROUPED "c" ("c","d")
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 3: Use the Alter Table statement to create a row storage column, which uses compression syntax.

```
gbase> DROP TABLE t2;
Query OK, 0 rows affected

gbase> CREATE TABLE t2 (a int, b varchar(10),c int ,d int);
Query OK, 0 rows affected

gbase> ALTER TABLE t2 ADD GROUPED c(c,d) COMPRESS('HighZ', 0);
Query OK, 0 rows affected
Records: 0    Duplicates: 0    Warnings: 0

gbase> show create table t2\G
*************************** 1. row ***************************
           Table: t2
Create Table: CREATE TABLE "t2" (
   "a" int(11) DEFAULT NULL,
   "b" int(11) DEFAULT NULL,
   "c" int(11) DEFAULT NULL,
   "d" int(11) DEFAULT NULL,
   GROUPED "c" ("c","d") COMPRESS('HighZ', 0)
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
```

```
1 row in set (Elapsed: 00:00:00.00)
```

5.1.8.9.4 Delete rows and columns when modifying tables

Grammar format

```
ALTER TABLE [[vc_name.]database_ name.] table_ name DROP GROUPED
grouped_ name;
```

Table -588 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table Name
grouped_ name	The name of the row or column.

Example

Example 1: Using the Alter Table statement to delete rows and columns.

```
gbase> ALTER TABLE t2 DROP GROUPED c;
Query OK, 0 rows affected
Records: 0    Duplicates: 0    Warnings: 0

gbase> SHOW CREATE TABLE t2 \G
*************************** 1. row ***************************
           Table: t2
Create Table: CREATE TABLE "t2" (
   "a" int(11) DEFAULT NULL,
   "b" int(11) DEFAULT NULL,
   "c" int(11) DEFAULT NULL,
   "d" int(11) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

5.1.8.10 Table level and column level cache loading and release

In order to improve query performance, GBase 8a MPP Cluster provides a management function that loads data directly into memory. Users can load all frequently used data from the entire table or specified columns into memory, and keep the loaded data in memory. This can reduce I/O operations, achieve direct access to table data from

memory, and improve query performance. For cluster products, the loading and release of cache are reflected in the gnode layer.

For the loading and releasing of table level and column level caches, there are mainly two states involved in the cache, namely the Locked state and KEEP state.

- Locked state DC: When accessing a DC that is being accessed by another thread or operator, the DC must first be locked (LOCK), and then unlocked (UNLOCK) after the access is completed.
- KEEP state DC: For frequently accessed tables, in order to improve access efficiency, all data (or data from a certain column) in the table is loaded into the cache using the Alter Table... CACHE command. When memory is low, cache swapping is not allowed for these DCs. KEEP state DCs can only be released through the Alter Table... NOCACHE command. KEEP status will cause data to reside in memory, reducing disk I/O operations.

For users, we recommend that the memory space occupied by loading table data should not exceed 50% of the available physical memory space.

5.1.8.10.1 Loading of memory

Grammar format

```
ALTER TABLE [vc_name.][database_name.]table_ name CACHE
[ (column_lists) ];
```

Table -589 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table Name
column_ lists	Column name, optional parameter. When specifying multiple columns, use English "," to separate them.

5.1.8.10.2 Release of memory

5.1.8.10.2.1 Release specified table memory

Function Description

Release all DCs in the cache (or specified table (tablename) that are not Locked and in a non KEEP state (loaded by the Alter Table CACHE command).

Grammar format

```
RELEASE CACHE [ON [vc_name.][database_name.]table_ name]
```

Table -590 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table Name

5.1.8.10.2.2 Release specified column memory

Function Description

Release all (or some columns) of non Locked memory in the specified table in the cache, including DCs in KEEP state (loaded by the Alter Table CACHE command).

Grammar format

```
ALTER TABLE [vc_name.][database_name.]table_ name NOCACHE
[ (column_lists) ]
```

Table -591 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table Name
column_ lists	Column name, optional parameter. When specifying multiple columns, use English "," to separate them.

Actions that affect KEEP status

The operations that affect the KEEP status are shown in the table below.

Table -592 Operations affecting KEEP status

Operations that affect KEEP	describe
DROP TABLE	The KEEP status data in the entire table is invalid.
ALTER TABLE ADD COLUMN	The KEEP status data in the entire table remains unchanged, and there is no KEEP status in the newly added column. It is necessary to re execute the Alter Table CACHE command to

Operations that affect KEEP	describe
	load.
ALTER TABLE DROP COLUMN	The KEEP status data of the column being DROP is invalid.
UPDATE TABLE	The KEEP status data of the updated column is invalid.

5.1.9 DML syntax

GBase 8a MPP Cluster supports standard DML statements.

5.1.9.1 INSERT

Function Description

Insert inserts a new row into an existing table. INSERT... The statements in the form of VALUES are inserted into record rows based on explicit values. INSERT ... A SELECT statement selects values from another table or tables and inserts them.

Grammar format

```
INSERT [INTO] [vc_name.][database_name.]table_ name [(col_name,...)]
VALUES ({expr | DEFAULT},...),(...),...
```

or

```
INSERT [INTO] [vc_name.][database_name.]table_ name [(col_name,...)]
SELECT ... FROM [vc_name.][database_name.]table_ name ...
```

Table -593 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional.
database_ name	Database name, optional.
table_ name	The table name is the table to be inserted into the data.
col_ name	Indicate which column the value specified by the statement is assigned to. If in Insert Values or Insert If the column column is not specified in the SELECT, then all column values must be in the VALUES() list or provided by the SELECT. If the user does not know the order of the columns in the table, they can use DESC table_ Name to view.

Parameter Name	explain
expr	Specify an expression expr to provide column values. For example, inserting an INT type data expression can be written as Insert INTO t (a) VALUES (3+5).
DEFAULT	Use the keyword DEFAULT to explicitly set the column as the default value. Using DEFAULT can avoid writing an incomplete list of VALUES that does not include all column values. DEFAULT (colname) can be used as a more general form of setting column defaults. If DEFAULT is not used, the user must specify the name of each column, corresponding to each value in VALUES.

explain

- If both the column list and the VALUES list are empty, Insert will create a row with each column set to its default value.

Example

Example 1: Insert INTO.

```
gbase> CREATE TABLE t0(id int) REPLICATED;
Query OK, 0 rows affected
gbase> INSERT INTO t0 VALUES(1),(2),(3),(4),(5),(6),(2),(3),(1);
Query OK, 9 rows affected
Records: 9  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t0;
+------+
| id   |
+------+
|    1 |
|    2 |
|    3 |
|    4 |
|    5 |
|    6 |
|    2 |
|    3 |
|    1 |
+------+
```

```
9 rows in set
```

Example 2: Insert INTO SELECT...。

```
gbase> CREATE TABLE t0(id int) REPLICATED;
Query OK, 0 rows affected

gbase> INSERT INTO t0(id) SELECT DISTINCT lo_ custkey FROM
ssbm.lineorder LIMIT 15;
Query OK, 15 rows affected
Records: 15  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t0;
+-------+
| id    |
+-------+
| 25738 |
| 18238 |
| 20612 |
|  5393 |
|  5954 |
| 11728 |
| 17302 |
| 14578 |
|  2210 |
| 27362 |
|  1642 |
| 29255 |
| 10745 |
|  7180 |
| 16276 |
+-------+
15 rows in set
```

Example 3: Insert INTO VALUES(DEFAULT)。

```
gbase> CREATE TABLE t0(id int DEFAULT 1) REPLICATED;
Query OK, 0 rows affected

gbase> INSERT INTO t0 (id) VALUES(DEFAULT);
Query OK, 1 row affected

--You can also specify default values for insertion in this way
gbase> INSERT INTO t0 (id) VALUES(DEFAULT(id));
Query OK, 1 row affected
```

```
gbase> SELECT * FROM t0;
+------+
| id   |
+------+
|    1 |
|    1 |
+------+
2 rows in set
```

Example 4: Automatically update the TIMESTAMP column when inserting. When using the Insert INTO t1 Values (), () syntax, the value of the TIMESTAMP column corresponding to the data row is automatically updated to a timestamp after the Insert is completed. The TIMESTAMP columns corresponding to b=6 and b=7 are automatically updated with a timestamp

```
DROP TABLE IF EXISTS t1;
CREATE TABLE t1(a timestamp , b int) DISTRIBUTED BY ('b');

gbase> INSERT INTO t1(b) VALUES(1);
Query OK, 1 row affected

gbase> INSERT INTO t1(b) VALUES (2);
Query OK, 1 row affected

gbase> INSERT INTO t1(b) VALUES (3);
Query OK, 1 row affected

gbase> INSERT INTO t1(b) VALUES (4);
Query OK, 1 row affected

gbase> INSERT INTO t1(b) VALUES (5);
Query OK, 1 row affected

gbase> SELECT * FROM t1 ORDER BY a;
+---------------------+------+
| a                   | b    |
+---------------------+------+
| 2013-12-17 14:04:43 |    1 |
| 2013-12-17 14:04:47 |    2 |
| 2013-12-17 14:04:52 |    3 |
| 2013-12-17 14:04:59 |    4 |
| 2013-12-17 14:05:04 |    5 |
+---------------------+------+
5 rows in set
```

```
gbase> INSERT INTO t1(b) VALUES (6),(7);
Query OK, 1 row affected

gbase> SELECT * FROM t1 ORDER BY a;
+---------------------+------+
| a                   | b    |
+---------------------+------+
| 2013-12-17 15:44:29 |    1 |
| 2013-12-17 15:44:30 |    2 |
| 2013-12-17 15:44:31 |    3 |
| 2013-12-17 15:44:32 |    4 |
| 2013-12-17 15:44:33 |    5 |
| 2013-12-17 15:44:48 |    7 |
| 2013-12-17 15:44:48 |    6 |
+---------------------+------+
7 rows in set
```

5.1.9.2 UPDATE

Function Description

When the value of the updated column is a valid expression, the correct update assignment operation can also be performed.

be careful

- The UPDATE operation does not support updating the value of the DISTRIBUTED BY column, the value of the auto increment column, and the value of the partition field in the partition table.

Grammar format

```
UPDATE [vc_name.][database_name.]table_ name
    SET col_ name1=expr1 [, col_name2=expr2 ...]
    [WHERE where_definition]
```

Table -594 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional.
database_ name	Database name, optional.

Parameter Name	explain
table_ name	The table name is the table to be inserted into the data.
col_ name	Indicate which column to update.
expr	Specify an expression expr to provide column values.

Example

First, create tables t0 and t2 and insert data.

```
gbase> CREATE TABLE t0(id int) REPLICATED;
Query OK, 0 rows affected

gbase> CREATE TABLE t2(id int);
Query OK, 0 rows affected

gbase> INSERT INTO t0(id) VALUES(1),(2),(3),(4),(5),(6),(2),(3),(1);
Query OK, 9 rows affected
Records: 9  Duplicates: 0  Warnings: 0

gbase> INSERT INTO t2(id) VALUES(1),(2),(4);
Query OK, 3 rows affected
Records: 3  Duplicates: 0  Warnings: 0
```

Example 1: Update the data of the t0 table.

```
gbase> SELECT * FROM t0;
+------+
| id   |
+------+
|    1 |
|    2 |
|    3 |
|    4 |
|    5 |
|    6 |
|    2 |
|    3 |
|    1 |
+------+
9 rows in set

gbase> UPDATE t0 SET t0.id = t0.id+1 WHERE t0.id > 1;
Query OK, 7 rows affected
Rows matched: 7  Changed: 7  Warnings: 0
```

```
gbase> SELECT * FROM t0;
+------+
| id   |
+------+
|    1 |
|    3 |
|    4 |
|    5 |
|    6 |
|    7 |
|    3 |
|    4 |
|    1 |
+------+
9 rows in set
```

Example 2: Multi table query update using IN.

```
gbase> SELECT * FROM t0;
+------+
| id   |
+------+
|    2 |
|    3 |
|    5 |
|    5 |
|    6 |
|    7 |
|    3 |
|    5 |
|    2 |
+------+
9 rows in set

gbase> UPDATE t0 SET t0.id = 10 WHERE t0.id IN (SELECT t2.id FROM
t2);
Query OK, 2 rows affected
Rows matched: 2  Changed: 2  Warnings: 0

gbase> SELECT * FROM t0;
+------+
| id   |
+------+
|   10 |
```

```
|      3 |
|      5 |
|      5 |
|      6 |
|      7 |
|      3 |
|      5 |
|     10 |
+------+
9 rows in set
```

Example 3: The sub query contains updated columns, and the update was successful.

```
gbase> CREATE TABLE t0(id int) REPLICATED;
Query OK, 0 rows affected
gbase> INSERT INTO t0(id) VALUES(1),(2),(3),(4),(5),(6),(2),(3),(1);
Query OK, 9 rows affected
Records: 9    Duplicates: 0    Warnings: 0

gbase> SELECT * FROM t0;
+------+
| id   |
+------+
|      1 |
|      2 |
|      3 |
|      4 |
|      5 |
|      6 |
|      2 |
|      3 |
|      1 |
+------+
9 rows in set

gbase> UPDATE t0 SET t0.id = (SELECT id FROM t0 WHERE id = 6);
Query OK, 9 rows affected
Rows matched: 9    Changed: 9    Warnings: 0
```

Example 4: Updating the value of the DISTRIBUTED BY column is not allowed.

```
gbase> CREATE TABLE student (stu_no int, stu_name varchar(200),stu_ sex
int) DISTRIBUTED BY('stu_no');
Query OK, 0 rows affected

gbase> INSERT INTO student (stu_no,stu_name,stu_sex) VALUES
```

```
(1,'Tom',0), (2,'Jim',0),(3,'Rose',1);
Query OK, 3 rows affected
Records: 3  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM student;
+--------+----------+---------+
| stu_ no | stu_ name | stu_ sex |
+--------+----------+---------+
|       1 | Tom        |        0 |
|       2 | Jim        |        0 |
|       3 | Rose       |        1 |
+--------+----------+---------+
3 rows in set

gbase> UPDATE student SET stu_ no = 4 WHERE stu_ no = 2;
ERROR 1722 (HY000): (GBA-02DD-0006) Can't update distributed column 'stu_
no'
```

Example 5: When updating, the numerical value of the TIMESTAMP column in the updated data row is automatically updated.

```
DROP TABLE IF EXISTS t2;
CREATE TABLE t2(a timestamp,b decimal(30,4),c int) DISTRIBUTED BY
('c');
INSERT INTO t2(b,c) VALUES(534.536,1);
INSERT INTO t2(b,c) VALUES(3534.56,11);
INSERT INTO t2(b,c) VALUES(33534.576,111);
INSERT INTO t2(b,c) VALUES(1334.56,1111);
INSERT INTO t2(b,c) VALUES(334.565,11111);
INSERT INTO t2(b,c) VALUES(34.5,111111);
INSERT INTO t2(b,c) VALUES(35.56,10009);
INSERT INTO t2(b,c) VALUES(3222.56,1897);
INSERT INTO t2(b,c) VALUES(3255.56,123);
INSERT INTO t2(b,c) VALUES(325.56,2);
gbase> SELECT * FROM t2 ORDER BY a;
+---------------------+------------+--------+
| a                   | b          | c      |
+---------------------+------------+--------+
| 2013-12-17 14:11:16 |  3534.5600 |     11 |
| 2013-12-17 14:11:16 | 33534.5760 |    111 |
| 2013-12-17 14:11:16 |  1334.5600 |   1111 |
| 2013-12-17 14:11:16 |   334.5650 |  11111 |
| 2013-12-17 14:11:16 |  3222.5600 |   1897 |
| 2013-12-17 14:11:16 |  3255.5600 |    123 |
| 2013-12-17 14:11:16 |   534.5360 |      1 |
```

```
| 2013-12-17 14:11:16 |       34.5000 | 111111 |
| 2013-12-17 14:11:16 |       35.5600 |    10009 |
| 2013-12-17 14:11:16 |      325.5600 |           2 |
+---------------------+---------------+--------+
10 rows in set

gbase> UPDATE t2 SET b=89.3 where c <1000;
Query OK, 5 rows affected
Rows matched: 5    Changed: 5    Warnings: 0


--Check the TIMESTAMP column corresponding to the data rows that have not been updated, and the value of the TIMESTAMP column remains unchanged.

gbase> SELECT * FROM t2 WHERE c <1000;
+---------------------+---------+------+
| a                   | b       | c    |
+---------------------+---------+------+
| 2013-12-17 14:17:57 | 89.3000 |     11 |
| 2013-12-17 14:17:57 | 89.3000 |    111 |
| 2013-12-17 14:17:57 | 89.3000 |    123 |
| 2013-12-17 14:17:57 | 89.3000 |      1 |
| 2013-12-17 14:17:57 | 89.3000 |      2 |
+---------------------+---------+------+
5 rows in set


--Check the TIMESTAMP column corresponding to the data rows that have not been updated, and the value of the TIMESTAMP column remains unchanged.

gbase> SELECT * FROM t2 WHERE c >=1000;
+---------------------+-----------+--------+
| a                   | b         | c      |
+---------------------+-----------+--------+
| 2013-12-17 14:11:16 | 1334.5600 |     1111 |
| 2013-12-17 14:11:16 |   334.5650 |    11111 |
| 2013-12-17 14:11:16 | 3222.5600 |     1897 |
| 2013-12-17 14:11:16 |      34.5000 | 111111 |
| 2013-12-17 14:11:16 |      35.5600 |    10009 |
+---------------------+-----------+--------+
5 rows in set
```

5.1.9.2.1 Quick UPDATE mode

Function Description

Quick UPDATE mode, which first deletes data that meets the update criteria, and then inserts new data that needs to be updated to the end of the table.

Compared to traditional row storage databases, updating a small number of rows with UPDATE in column storage data is relatively time-consuming. Therefore, GBase 8a MPP Cluster has specially designed a fast UPDATE mode to improve data update operations.

The fast UPDATE mode currently only supports operations on table objects.

To use fast UPDATE mode, SET gbase must be used on the client side_ fast_ update =1; The command opens the fast UPDATE mode. It is recommended to use the default UPDATE mode when updating bulk data, and the fast UPDATE mode when updating small amounts of data.

SET gbase_ fast_ update =0; Indicates that fast UPDATE mode is turned off.

SET gbase_ fast_ update =1; Indicates that fast UPDATE mode is enabled.

Example

Example 1: Enable fast UPDATE mode.

```
gbase> CREATE TABLE t1 (f_1 int);
Query OK, 0 rows affected

gbase> INSERT INTO t1 values(1),(2),(3);
Query OK, 3 rows affected
Records: 3    Duplicates: 0    Warnings: 0

gbase> SELECT * FROM t1;
+------+
| f_1  |
+------+
|    1 |
|    2 |
|    3 |
+------+
3 rows in set

gbase> SET gbase_ fast_ update = 1;
Query OK, 0 rows affected

gbase> UPDATE t1 SET f_ 1 = 10 WHERE f_ 1= 1;
Query OK, 1 row affected
Rows matched: 1    Changed: 1    Warnings: 0

gbase> SELECT * FROM t1;
```

```
+------+
| f_1  |
+------+
|    2 |
|    3 |
|   10 |
+------+
3 rows in set

gbase> SET gbase_ fast_ update = 0;
Query OK, 0 rows affected
```

5.1.9.2.2 UPDATE From statement

Function Description

After using the update from statement, the target table target_ table_ The column values in the rows with corresponding relationships in name1 are referenced by the source table source_ table_ In name2, the column value or expression value of the corresponding row is updated, and there is no update for rows without corresponding relationships.

be careful

- Updated target table target_ table_ For each row of data in name1, there can be at most one row of source table that meets the where update criteria_ table_ Name2 data, that is, according to where update conditions, table target_ table_ Record and Table Source in name1_ table_ The corresponding relationship of records in name2 is 1 to 1, many to 1, or 1 to 0.
- Unsupported scenario is the updated target table target_ table_ One or more rows of data in name1 have one or more source tables that meet the where update criteria_ table_ Name2 data, that is, according to where update conditions, table target_ table_ Record and Table Source in name1_ table_ The correspondence between records in name2 is 1-to-many.

Grammar format

Update *target_ table_ name1* from *source_ table_ name2* set *col_ name1=expr1[,col_name2=expr2...]*
[where *where_definition*]

Table 597 Parameter Description

Parameter Name	explain

Parameter Name	explain
target_ table_ name1	The name of the target table being updated.
source_ table_ name2	Source table table name.
col_ name1	Indicate which column to update.
expr1	Specify an expression expr1 to provide column values.
where_ definition	Update conditions.

Example

```
gbase> select * from t1;
+------+-------+
| id   | name  |
+------+-------+
|    1 | allen |
|    2 | Tom   |
|    3 | carl  |
|    4 | lily  |
|    5 | jim   |
+------+-------+
5 rows in set (Elapsed: 00:00:00.03)

gbase> select * from t2;
+------+-----------+------+
| id   | name      | age  |
+------+-----------+------+
|    1 | Allen2000 |   10 |
|    2 | Tom2000   |   12 |
|    3 | Carl2000  |   11 |
+------+-----------+------+
3 rows in set (Elapsed: 00:00:00.01)

gbase> select * from t3;
+------+-----------+------+
| id   | name      | age  |
+------+-----------+------+
|    1 | Allen2000 |   10 |
|    1 | Tom2000   |   12 |
|    3 | Carl2000  |   11 |
+------+-----------+------+
3 rows in set (Elapsed: 00:00:00.02)

gbase> update t1 from t2 set name=t2.name where t1.id=t2.id;
```

```
Query OK, 3 rows affected (Elapsed: 00:00:00.12)
Rows matched: 3  Changed: 3  Warnings: 0

gbase> select * from t1;
+------+-----------+
| id   | name      |
+------+-----------+
|    1 | Allen2000 |
|    2 | Tom2000   |
|    3 | Carl2000  |
|    4 | lily      |
|    5 | jim       |
+------+-----------+
5 rows in set (Elapsed: 00:00:00.03)

gbase> update t1 from t3 set name=t3.name where t1.id=t3.id;
ERROR 1709 (HY000): No successful nodes for suffix:n1, cause: [172.16.79.11:5050](GBA-02AD-0005)Failed to query in gnode:
DETAIL: (GBA-01EX-700) Gbase general error: can not update one row to multi-data
SQL: /*172.16.79.11_ 15_ 22_ 2023-05-04_ 17:23:29*/ UPDATE /*+ TID('524837') */ `gctmpdb`._ tmp_ 189730988_ 15_ t6_ 1_ 1682319836_ s INNER JOIN `test`.` t1_ n1` `vcname000001.test.t1` ON (`vcname000001.test.t1`.`id` = `_tmp_189730988_15_t6_1_1682319836_s`.`id`) SET `vcname000001.test.t1`.`name` = `_ tmp_ 189730988_ 15_ t6_ 1_ 1682319836_ s`.` vcn
```

5.1.9.3 DELETE

Grammar format

```
DELETE [FROM] [vc_name.][database_name.]table_ name [tbl_alias] [WHERE where_definition]
```

Table 595 Parameter Description

Parameter Name	explain
vc_ name	Virtual cluster name, optional.
database_ name	Database name, optional.
table_ name	The table name is the table to be inserted into the data.
tbl_ alias	alias


explain

- When the DELETE statement contains aliases, the From keyword can be omitted.

Example

Tables and data used in the example:

```
CREATE TABLE t0 (id int);
INSERT INTO t0 values(1),(2),(3),(4),(5),(6),(7),(8);
```

Example 1: Delete data with id greater than 6 in the table.

```
gbase> DELETE FROM t0 WHERE t0.id > 6;
Query OK, 2 rows affected
```

Example 2: Using IN, delete data with id values of 1, 2, and 3.

```
gbase> DELETE FROM t0 WHERE t0.id IN ( 1,2,3);
Query OK, 3 rows affected
```

Example 3: Delete full table data.

```
gbase> DELETE FROM t0;
Query OK, 3 rows affected
```

Example 4: DELETE From WHERE... IN (SELECT...FROM)。

```
gbase> INSERT INTO t0 values(1),(2),(3),(4),(5),(6),(7),(8);
Query OK, 8 rows affected
Records: 8  Duplicates: 0  Warnings: 0

gbase> DELETE FROM t0 WHERE t0.ID IN (SELECT id FROM t0);
Query OK, 8 rows affected
```

Example 5: The DELETE syntax contains aliases for tables, and the From keyword can be omitted.

```
gbase> INSERT INTO t0 values(1),(2),(3),(4),(5),(6),(7),(8);
Query OK, 8 rows affected
Records: 8  Duplicates: 0  Warnings: 0

gbase> DELETE FROM t0 tt WHERE tt.id=8;
Query OK, 1 row affected

gbase> DELETE t0 tt WHERE tt.id=1;
Query OK, 1 row affected
```

```
gbase> SELECT * FROM t0;
+------+
| id   |
+------+
|    2 |
|    3 |
|    4 |
|    5 |
|    6 |
|    7 |
+------+
6 rows in set
```

Example 6: DELETE WHERE...

```
gbase> DELETE t0 WHERE id = 2;
Query OK, 1 row affected
```

5.1.9.4 MERGE

Grammar format

```
MERGE [INTO] [vc_name.][database_name.]table_ name
USING table_ reference
ON conditional_ exp
[WHEN MATCHED THEN UPDATE SET col_name1=expr1 [, col_name2=expr2] ...
[WHEN NOT MATCHED THEN INSERT [(col_name3,...)] VALUES (expr3,...)]
```

Example statements using MERGE syntax are as follows:

```
MERGE INTO a
USING table_ x b
ON (a.hash_col = b.hash_col)
WHEN MATCHED THEN UPDATE SET a.column = b.column,…
WHEN NOT MATCHED THEN INSERT (a.column,…) VALUES (b.column,…)
```

Table 596 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	The table name must be a table, not a view, and an alias can be used.

Parameter Name	explain
table_ reference	table_ Reference is only allowed as a table and can be aliased.
conditional_ exp	Associated conditions
column	Column Name

be careful

- The UPDATE section does not support the DELETE clause.
- The UPDATE and Insert sections do not support the WHERE clause.
- The UPDATE part and INSERT part can be omitted, but not both, otherwise syntax error will be reported.
- The positions of the UPDATE and Insert sections cannot be reversed.
- The VALUES section of Insert does not allow the use of MERGE tables.
- If a column in UPDATE or Insert appears multiple times, there will be no error and the specified column will take effect. However, it is recommended not to rely on this behavior to avoid such use.
- One to many updates are not allowed: If one row in the MERGE table meets the join criteria with multiple rows in the USING table, an error is reported.
- In the MERGE INTO a statement, table a must be a hash distribution table.
- USING table_ The section on x b ON (a.hash_col=b. hash_col) contains a JOIN conditional statement. The JOIN condition must have an equivalent correlation condition for the hash distribution column of a table, and under this correlation condition, the hash distribution column must be a physical column association and cannot be an expression or function.
- For example:
- a.hash_ col = b.hash_ Col (legal)
- ABS(a.hash_col) = b.hash_ Col (illegal)
- a.hash_ Col=ABS (b. hash_col) (illegal), which is due to the limitations of current cluster hash redistribution. JOIN uses physical columns for hash redistribution.
- When MATCHED THE UPDATE SET a.column=b. column, in the... statement, the set column (a.column) cannot be a hash distribution column.
- In the WHERE NOT MATCHED THEN Insert (a.column,...) VALUES (b. column,...) statement, in the field list of the Insert, the a.column must have a hash distribution column, and in the corresponding column of VALUES, the b. column must be a hash distribution column, or a hash distribution column after dynamic hash redistribution.
- If the JOIN relationship between table a and table b is not a static hash distribution JOIN relationship, then gcluster_ hash_ redistribute_ join_ The optimize parameter cannot be turned off.
- If the main shard of a table participating in MERGE operations is in a locked state, hash redistribution cannot be used. Therefore, when a and b in the above example are not in a static hash JOIN relationship, MERGE cannot be executed.

Example

Example 1: MERGE the t1 table.

```
DROP TABLE IF EXISTS t1;
CREATE TABLE t1(i int,vc varchar(20),d date,dc decimal(20,3)) DISTRIBUTED BY ('i');
INSERT INTO t1 VALUES(1,'one','2013-02-03',11.21);
INSERT INTO t1 VALUES (2,'two','2013-04-03',12.21);
INSERT INTO t1 VALUES (3,'one2','2013-03-03',31.21);
INSERT INTO t1 VALUES (11,'one3','2013-08-03',41.21);
INSERT INTO t1 VALUES (14,'three','2013-07-22',161.218);
INSERT INTO t1 VALUES (33,'third','2013-09-04',11.216);
INSERT INTO t1 VALUES (5,'wto','2013-02-03',110.210);
INSERT INTO t1 VALUES (null,'first','2013-02-03',311.91);
INSERT INTO t1 VALUES (8,'five','2013-02-03',811.201);

DROP TABLE IF EXISTS t2;
CREATE TABLE t2(i int,vc varchar(20),d date,dc decimal(30,3)) DISTRIBUTED BY ('i');
INSERT INTO t2 VALUES (1,'one','2013-02-03',11.20);
INSERT INTO t2 VALUES (2,'two','2013-08-03',12.81);
INSERT INTO t2 VALUES (13,'one2','2013-09-03',31.01);
INSERT INTO t2 VALUES (110,'one3','2013-08-03',41.21);
INSERT INTO t2 VALUES (14,'three','2013-06-22',161.218);
INSERT INTO t2 VALUES (30,'third','2013-09-04',11.216);

gbase> SELECT * FROM t1 ORDER BY i;
+------+-------+------------+---------+
| i    | vc    | d          | dc      |
+------+-------+------------+---------+
|    1 | one   | 2013-02-03 |  11.210 |
|    2 | two   | 2013-04-03 |  12.210 |
|    3 | one2  | 2013-03-03 |  31.210 |
|    5 | wto   | 2013-02-03 | 110.210 |
|    8 | five  | 2013-02-03 | 811.201 |
|   11 | one3  | 2013-08-03 |  41.210 |
|   14 | three | 2013-07-22 | 161.218 |
|   33 | third | 2013-09-04 |  11.216 |
| NULL | first | 2013-02-03 | 311.910 |
+------+-------+------------+---------+
9 rows in set

gbase> SELECT * FROM t2 ORDER BY i;
+------+-------+------------+---------+
| i    | vc    | d          | dc      |
+------+-------+------------+---------+
```

```
|      1 | one    | 2013-02-03 |   11.200 |
|      2 | two    | 2013-08-03 |   12.810 |
|     13 | one2   | 2013-09-03 |   31.010 |
|     14 | three | 2013-06-22 | 161.218 |
|     30 | third | 2013-09-04 |   11.216 |
|    110 | one3   | 2013-08-03 |   41.210 |
+------+-------+------------+----------+
6 rows in set

gbase> MERGE INTO t1 USING t2 ON t1.i=t2.i WHEN MATCHED THEN UPDATE SET t1.vc=t2.vc WHEN NOT MATCHED THEN INSERT(t1.i,t1.vc) VALUES(t2.i,t2.vc);
Query OK, 6 rows affected
Rows matched: 6    Changed: 6    Warnings: 0

gbase> SELECT * FROM t1 ORDER BY i;
+------+-------+------------+----------+
| i    | vc    | d          | dc       |
+------+-------+------------+----------+
|      1 | one    | 2013-02-03 |   11.210 |
|      2 | two    | 2013-04-03 |   12.210 |
|      3 | one2   | 2013-03-03 |   31.210 |
|      5 | wto    | 2013-02-03 | 110.210 |
|      8 | five   | 2013-02-03 | 811.201 |
|     11 | one3   | 2013-08-03 |   41.210 |
|     13 | one2   | NULL       |     NULL |
|     14 | three | 2013-07-22 | 161.218 |
|     30 | third | NULL       |     NULL |
|     33 | third | 2013-09-04 |   11.216 |
|    110 | one3   | NULL       |     NULL |
| NULL | first | 2013-02-03 | 311.910 |
+------+-------+------------+----------+
12 rows in set
```

Example 2: Table t1 uses the alias t and then performs a MERGE operation.

```
DROP TABLE IF EXISTS t1;
CREATE TABLE t1(i int,vc varchar(20),d date,dc decimal(20,3)) DISTRIBUTED BY ('i');
INSERT INTO t1 VALUES(1,'one','2013-02-03',11.21);
INSERT INTO t1 VALUES (2,'two','2013-04-03',12.21);
INSERT INTO t1 VALUES (3,'one2','2013-03-03',31.21);
INSERT INTO t1 VALUES (11,'one3','2013-08-03',41.21);
INSERT INTO t1 VALUES (14,'three','2013-07-22',161.218);
INSERT INTO t1 VALUES (33,'third','2013-09-04',11.216);
```

```
INSERT INTO t1 VALUES (5,'wto','2013-02-03',110.210);
INSERT INTO t1 VALUES (null,'first','2013-02-03',311.91);
INSERT INTO t1 VALUES (8,'five','2013-02-03',811.201);

DROP TABLE IF EXISTS t2;
CREATE TABLE t2(i int,vc varchar(20),d date,dc decimal(30,3))
DISTRIBUTED BY ('i');
INSERT INTO t2 VALUES (1,'one','2013-02-03',11.20);
INSERT INTO t2 VALUES (2,'two','2013-08-03',12.81);
INSERT INTO t2 VALUES (13,'one2','2013-09-03',31.01);
INSERT INTO t2 VALUES (110,'one3','2013-08-03',41.21);
INSERT INTO t2 VALUES (14,'three','2013-06-22',161.218);
INSERT INTO t2 VALUES (30,'third','2013-09-04',11.216);
gbase> SELECT * FROM t1 ORDER BY i;
+------+-------+------------+---------+
| i    | vc    | d          | dc      |
+------+-------+------------+---------+
|    1 | one   | 2013-02-03 |  11.210 |
|    2 | two   | 2013-04-03 |  12.210 |
|    3 | one2  | 2013-03-03 |  31.210 |
|    5 | wto   | 2013-02-03 | 110.210 |
|    8 | five  | 2013-02-03 | 811.201 |
|   11 | one3  | 2013-08-03 |  41.210 |
|   14 | three | 2013-07-22 | 161.218 |
|   33 | third | 2013-09-04 |  11.216 |
| NULL | first | 2013-02-03 | 311.910 |
+------+-------+------------+---------+
9 rows in set

gbase> SELECT * FROM t2 ORDER BY i;
+------+-------+------------+---------+
| i    | vc    | d          | dc      |
+------+-------+------------+---------+
|    1 | one   | 2013-02-03 |  11.200 |
|    2 | two   | 2013-08-03 |  12.810 |
|   13 | one2  | 2013-09-03 |  31.010 |
|   14 | three | 2013-06-22 | 161.218 |
|   30 | third | 2013-09-04 |  11.216 |
|  110 | one3  | 2013-08-03 |  41.210 |
+------+-------+------------+---------+
6 rows in set
gbase> MERGE INTO t1 t USING t2 ON t.i=t2.i WHEN MATCHED THEN
UPDATE SET t.vc=t2.vc WHEN NOT MATCHED THEN INSERT (t.i,t.vc)
VALUES (t2.i,t2.vc);
```

```
Query OK, 6 rows affected
Rows matched: 6    Changed: 6    Warnings: 0

gbase> SELECT * FROM t1 ORDER BY i;
+------+-------+------------+---------+
| i       | vc        | d                   | dc            |
+------+-------+------------+---------+
|       1 | one      | 2013-02-03 |    11.210 |
|       2 | two      | 2013-04-03 |    12.210 |
|       3 | one2    | 2013-03-03 |    31.210 |
|       5 | wto      | 2013-02-03 | 110.210 |
|       8 | five     | 2013-02-03 | 811.201 |
|      11 | one3    | 2013-08-03 |    41.210 |
|      13 | one2    | NULL              |        NULL |
|      14 | three | 2013-07-22 | 161.218 |
|      30 | third | NULL              |        NULL |
|      33 | third | 2013-09-04 |    11.216 |
|    110 | one3    | NULL              |        NULL |
| NULL | first | 2013-02-03 | 311.910 |
+------+-------+------------+---------+
12 rows in set
```

Example 3: After the MERGE operation, synchronously update the value of the TIMESTAMP column.

```
DROP TABLE IF EXISTS t1;
CREATE TABLE t1(a timestamp ,b int) DISTRIBUTED BY('b');
INSERT INTO t1(b) VALUES(1);
INSERT INTO t1(b) VALUES(2);
INSERT INTO t1(b) VALUES(6);
INSERT INTO t1(b) VALUES(8);
INSERT INTO t1(b) VALUES(107);
INSERT INTO t1(b) VALUES(105);

gbase> SELECT * FROM t1 ORDER BY a;
+---------------------+------+
| a                              | b       |
+---------------------+------+
| 2013-12-17 14:40:03 |       6 |
| 2013-12-17 14:40:03 |       8 |
| 2013-12-17 14:40:03 |    105 |
| 2013-12-17 14:40:03 |       1 |
| 2013-12-17 14:40:03 |       2 |
| 2013-12-17 14:40:03 |    107 |
+---------------------+------+
```

```
6 rows in set

gbase> SELECT * FROM tt ORDER BY a;
+---------------------+------+------+
| a                   | b    | c    |
+---------------------+------+------+
| 2013-12-17 14:40:44 |  105 | a    |
+---------------------+------+------+
1 rows in set

gbase> MERGE INTO tt USING t1 ON t1.b=tt.b WHEN MATCHED THEN UPDATE SET tt.c='b' WHEN NOT MATCHED THEN INSERT (tt.b) VALUES(t1.b);
Query OK, 6 rows affected
Rows matched: 6  Changed: 6  Warnings: 0

--Looking at the data in table t1, the TIMESTAMP column was not updated synchronously.

gbase> SELECT * FROM t1 ORDER BY a;
+---------------------+------+
| a                   | b    |
+---------------------+------+
| 2013-12-17 14:40:03 |    6 |
| 2013-12-17 14:40:03 |    8 |
| 2013-12-17 14:40:03 |  105 |
| 2013-12-17 14:40:03 |    1 |
| 2013-12-17 14:40:03 |    2 |
| 2013-12-17 14:40:03 |  107 |
+---------------------+------+
6 rows in set

--View the data in the tt table and synchronize the TIMESTAMP column updates.

gbase> SELECT * FROM tt ORDER BY a;
+---------------------+------+------+
| a                   | b    | c    |
+---------------------+------+------+
| 2013-12-17 14:40:20 |    8 | NULL |
| 2013-12-17 14:40:20 |    1 | NULL |
| 2013-12-17 14:40:20 |  105 | b    |
| 2013-12-17 14:40:20 |    6 | NULL |
| 2013-12-17 14:40:20 |    2 | NULL |
| 2013-12-17 14:40:20 |  107 | NULL |
+---------------------+------+------+
```

```
6 rows in set
```

5.1.9.5 DML of the specified partition name in the partition table

Sample Use Case Table

```
CREATE TABLE "pt" (
   "i" int(11) DEFAULT NULL,
   "j" int(11) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_ tablespace'
 PARTITION BY LIST (mod(i,2))
(PARTITION p0 VALUES IN (0) TABLESPACE = 'sys_ tablespace' ENGINE = EXPRESS,
 PARTITION p1 VALUES IN (1) TABLESPACE = 'sys_ tablespace' ENGINE = EXPRESS)
```

Syntax SELECT

```
SELECT ... FROM [vc_name.][database_name.]<table_ name> [PARTITION (partition_name1[, partition_name2, ...])]
```

Example:

```
gbase> create table tf(a int,d int)
partition by range (d)
        (
        partition p0 values less than (1990),
        partition p1 values less than (2000),
        partition p2 values less than (2010),
        partition p3 values less than (2020),
        partition p4 values less than (2030)
        );
gbase> select * from tf partition(p4);
```

```
+------+------+
| a    | d    |
+------+------+
|    1 | 2023 |
+------+------+
1 row in set (Elapsed: 00:00:00.01)
```

Syntax DELETE

```
DELETE ... FROM [vc_name.][database_name.]<table_ name> [PARTITION
(partition_name1[, partition_name2, ...])]
```

Example:

```
gbase> insert into pt values(1,1),(2,2);
Query OK, 2 rows affected (Elapsed: 00:00:01.70)
gbase> delete from pt partition(p1);
Query OK, 1 row affected (Elapsed: 00:00:03.27)
gbase> delete from pt partition(p0,p1);
Query OK, 1 row affected (Elapsed: 00:00:00.11)
```

Syntax UPDATE

```
UPDATE [vc_name.][database_name.]<table_ name> [PARTITION
(partition_name1[, partition_name2, ...])] set…
```

Description: Partition condition column does not allow updates

Example:

```
gbase> insert into pt values(1,1),(2,2);
Query OK, 2 rows affected (Elapsed: 00:00:01.00)
Update specified partition data
gbase> update pt partition (p0) set j=j+1;
Query OK, 1 row affected (Elapsed: 00:00:02.41)
```

```
gbase> select * from pt;
    +------+------+
    | i    | j    |
    +------+------+
    |    2 |    3 |
    |    1 |    1 |
    +------+------+
2   rows in set (Elapsed: 00:00:01.20)
gbase> update pt partition (p0,p1) set j=j+1;
Query OK, 2 rows affected (Elapsed: 00:00:03.09)
Rows matched: 2  Changed: 2  Warnings: 0
gbase> select * from pt;
    +------+------+
    | i    | j    |
    +------+------+
    |    2 |    4 |
    |    1 |    2 |
    +------+------+
3   rows in set (Elapsed: 00:00:00.47)
```

Update the data of the specified partition in fast update mode

```
gbase> set gbase_ fast_ update=1;
Query OK, 0 rows affected (Elapsed: 00:00:01.39)
gbase> update pt partition (p0) set j=j+1;
Query OK, 1 row affected (Elapsed: 00:00:14.89)
```

5.1.10 DQL syntax

5.1.10.1 SELECT

Grammar format

```
SELECT
        [ALL | DISTINCT | DISTINCTROW ]
      select_ expr, ...
      [FROM table_references]
            [WHERE where_definition]
        [GROUP BY {col_name | expr | position}
          , ...]
        [QUALIFY search_conditions]
        [HAVING where_definition]
            [ORDER BY {col_name | expr | position}
          [ASC | DESC] , ...]
        [LIMIT {[offset,] row_count | row_count OFFSET offset}]
        [INTO OUTFILE 'file_name' export_options]
```

Table 597 Parameter Description

Parameter Name	explain
ALL, DISTINCT, and DISTINCTROW	Specify whether to return duplicate rows, default to ALL (all matching rows are returned). DISTINCT and DISTINCTROW are synonymous and are used to remove duplicate rows from the result set.
select_ expr	Refers to the columns displayed in the query, and AS can be used to name aliases for the columns displayed in SELECT. The aliases should not be duplicated with the column names displayed in SELECT.
table_ references	Specify one or more tables from which to find travel. Its syntax is described in JOIN syntax.
where_ definition	The expression containing the WHERE keyword, followed by one or more conditions to be satisfied by the query, but the aggregate function cannot be used in where.
[GROUP BY {col_name | expr |	Refer to the "GROUP BY" section.

Parameter Name	explain
position}, ...] [HAVING where_definition] | [QUALIFY *search_conditions*]	HAVING and QUALIFY syntax cannot appear simultaneously.
[ORDER BY {col_name | expr | position} [ASC | DESC] , ...]	Refer to the "ORDER BY" section.
[LIMIT {[offset,] row_count | row_count OFFSET offset}]	Refer to the "LIMITED" section.
[INTO OUTFILE 'file_name' export_options]	Refer to the section 'SELECT INTO OUTFILE'.

5.1.10.1.1JOIN

Function Description

JOIN refers to querying data in two or more tables.

The JOIN functions supported by GBase 8a MPP Cluster can be roughly divided into the following categories:

- INNER JOIN: Retrieves records of matching relationships between fields in two tables. In the absence of connection conditions, INNER JOIN and (comma) are semantically equivalent: they generate a Cartesian product between the specified two tables (that is, each row in the first table is connected to all rows in the second table).
- Left JOIN: Retrieves all records in the left table, even if there are no corresponding matching records in the right table. If there are no matching records in the right table of the ON or USING section of the Left JOIN, then all columns in one row of the right table are set to NULL. This method can be used to identify records that do not match between one table and another.
- Right JOIN: Contrary to Left JOIN, used to retrieve all records in the right table,

even if there are no corresponding matching records in the left table.

- FULL JOIN/FULL OUTER JOIN: can be seen as a combination of left JOIN and right JOIN results. But FULL JOIN and Left JOIN UNION RIGHT JOIN... Not completely equivalent, as UNION removes duplicate records.

explain

- When performing a simple equivalent association (which requires both the equivalent JOIN and the equivalent JOIN conditions to be physical columns on both sides rather than expressions), and the JOIN involves distributed Hash JOIN, the following constraints need to be met:
  - The data types on both sides of JOIN must be the same;
  - When conducting CHAR JOIN CHAR, the accuracy must be the same;
  - The data types on both sides of JOIN are different, and only CHAR and VARCHAR are supported. The INT family (including INT, BIGINT, SMALLINT, TINYINT) and VARCHAR are associated for querying.

Grammar format

GBase 8a MPP GCluster SQL supports the following JOIN syntax:

```
table_ references:
      table_ reference [, table_reference] …

table_ reference:
      table_ factor
  | join_ table

table_ factor:
       [vc_name.][database_name.]table_ name [[AS] alias]
  | ( table_references )

join_ table:
    table_ reference [INNER | CROSS] JOIN table_ factor [join_condition]
  | table_ reference LEFT [OUTER] JOIN table_ reference join_ condition
  | table_ reference RIGHT [OUTER] JOIN table_ reference join_ condition
  | table_ reference FULL [OUTER] JOIN table_ reference join_ condition

join_ condition:
      ON conditional_ expr
  | USING (column_list)
```

Table -598 Parameter Description

Parameter Name	explain
USING(column_list)	Used to name a series of columns. These columns must exist in both tables simultaneously. If both tables a and b contain columns c1, c2, and c3, the following union will contain corresponding columns from both tables: 1）a LEFT JOIN b USING (c1,c2,c3) 2）a LEFT JOIN b ON a.c1 = b.c1 AND a.c2 = b.c2 AND a.c3 = b.c3
Vc_ name	VC command, optional.
database_ name	Database name, optional.
table_ name	Table Name

Example

Example 1: Example of join syntax.

Tables and data used in the example:

```
CREATE TABLE t1(id INT,name VARCHAR(30));
CREATE TABLE t2(id INT,title VARCHAR(20),name VARCHAR(30));
INSERT INTO t1 VALUES(1,'name1'),(2,'name2'),(3,'name3');
INSERT INTO t2 VALUES(1,'t1','name1'),(3,'t3','name3'),(4,'t4','name4');

gbase> SELECT * FROM t1;
+------+-------+
| id   | name  |
+------+-------+
|    1 | name1 |
|    2 | name2 |
|    3 | name3 |
+------+-------+
3 rows in set (Elapsed: 00:00:00.07)

gbase> SELECT * FROM t2;
+------+-------+-------+
| id   | title | name  |
+------+-------+-------+
|    1 | t1    | name1 |
```

```
|       3 | t3       | name3 |
|       4 | t4       | name4 |
+-------+--------+-------+
3 rows in set (Elapsed: 00:00:00.03)
```

INNER JOIN:

gbase> SELECT a.id,a.name,b.name,b.title FROM t1 a INNER JOIN t2 b ON a.id=b.id;

```
+------+-------+-------+-------+
| id   | name  | name  | title |
+------+-------+-------+-------+
|    1 | name1 | name1 | t1    |
|    3 | name3 | name3 | t3    |
+------+-------+-------+-------+
2 rows in set (Elapsed: 00:00:00.15)
```

The WHERE clause method is equivalent to the INNER JON above:

gbase> SELECT a.id,a.name,b.name,b.title FROM t1 a,t2 b WHERE a.id=b.id;

```
+------+-------+-------+-------+
| id   | name  | name  | title |
+------+-------+-------+-------+
|    1 | name1 | name1 | t1    |
|    3 | name3 | name3 | t3    |
+------+-------+-------+-------+
2 rows in set (Elapsed: 00:00:00.12)
```

LEFT JOIN:

gbase> SELECT a.id,a.name,b.name,b.title FROM t1 a LEFT JOIN t2 b ON a.id=b.id;

```
+------+-------+-------+-------+
| id   | name  | name  | title |
+------+-------+-------+-------+
|    2 | name2 | NULL  | NULL  |
|    1 | name1 | name1 | t1    |
|    3 | name3 | name3 | t3    |
+------+-------+-------+-------+
3 rows in set (Elapsed: 00:00:00.13)
```

RHGHT JOIN:

gbase> SELECT a.id,a.name,b.name,b.title FROM t1 a RIGHT JOIN t2 b ON a.id=b.id;

```
+------+-------+-------+-------+
| id   | name  | name  | title |
+------+-------+-------+-------+
| NULL | NULL  | name4 | t4    |
|    1 | name1 | name1 | t1    |
|    3 | name3 | name3 | t3    |
+------+-------+-------+-------+
3 rows in set (Elapsed: 00:00:00.16)
```

FULL JOIN:

```
gbase> SELECT a.id,a.name,b.name,b.title FROM t1 a FULL JOIN t2 b ON
a.id=b.id;
+------+-------+-------+-------+
| id   | name  | name  | title |
+------+-------+-------+-------+
|    2 | name2 | NULL  | NULL  |
| NULL | NULL  | name4 | t4    |
|    1 | name1 | name1 | t1    |
|    3 | name3 | name3 | t3    |
+------+-------+-------+-------+
4 rows in set (Elapsed: 00:00:00.17)
```

Example 2: char JOIN char (same precision) example.

Tables and data used in the example:

```
CREATE TABLE t3(a CHAR(5), b CHAR(10));
CREATE TABLE t4(a CHAR(10));
INSERT INTO t3 VALUES('abcde', 'abcde');
INSERT INTO t4 VALUES('abcde');

gbase> SELECT * FROM t3;
+-------+------------+
| a     | b          |
+-------+------------+
| abcde | abcde      |
+-------+------------+
1 row in set (Elapsed: 00:00:00.03)

gbase> SELECT * FROM t4;
+------------+
| a          |
+------------+
| abcde      |
```

```
+------------+
1 row in set (Elapsed: 00:00:00.02)
```

Both t3. b and t4. a types are char (10), so an equivalent JOIN relationship can be established:

```
gbase>    SELECT * FROM t3 JOIN t4 ON t3.b = t4.a;
+-------+------------+------------+
| a     | b          | a          |
+-------+------------+------------+
| abcde | abcde      | abcde      |
+-------+------------+------------+
1 row in set (Elapsed: 00:00:00.06)
```

The t3. a type is char (5), and the t4. a type is char (10). It is prohibited to establish equivalent JOIN relationships between char types with different accuracies:

```
gbase> SELECT * FROM t3 JOIN t4 ON t3.a = t4.a;
ERROR 1149 (42000): (GBA-02SC-1001) Data types of equivalence join relation ((`vc1.demo.t3`.`a` = `vc1.demo.t4`.`a`)) are not supported , data types: left is CHAR(5), right is CHAR(10)
```

Example 3: char JOIN varchar example.

Tables and data used in the example:

```
CREATE TABLE t1(a CHAR(5), b CHAR(10));
CREATE TABLE t2(a varchar(10));
INSERT INTO t1 VALUES('abcde', 'abcde');
INSERT INTO t2 VALUES('abcde');
```

When T1. a type is char (5), 'abcde' does not need to be supplemented with spaces, and is equivalent to 'abcde' of varchar:

```
gbase> SELECT * FROM t1 JOIN t2 ON t1.a = t2.a;

+-------+------------+-------+
| a     | b          | a     |
+-------+------------+-------+
| abcde | abcde      | abcde |
+-------+------------+-------+
1 row in set
```

The type t1. b is char (10), and after filling in spaces, it is' abcde ',

which is different from the' abcde 'of varchar, and the following result set is empty:

```
gbase> SELECT * FROM t1 JOIN t2 ON t1.b = t2.a;
Empty set
```

Example 4: Example of int JOIN varchar.

Tables and data used in the example:

```
CREATE TABLE t5(a INT);
CREATE TABLE t6(a VARCHAR(10));
INSERT INTO t5 VALUES(179);
INSERT INTO t6 VALUES('179');
```

An equivalent JOIN relationship can be established between int and varchar:

```
gbase> SELECT * FROM t5 JOIN t6 ON t5.a = t6.a;
+------+------+
| a    | a    |
+------+------+
|  179 | 179  |
+------+------+
1 row in set
```

5.1.10.1.2 GROUP BY ...

Function Description

Divide a data set into several small areas (i.e. groups) through certain rules, and then process data for several small areas. Generally, GROUP BY is used together with aggregate function and OLAP function.

Grammar format

```
GROUP BY {col_name | expr | position}, ... [HAVING where_definition] | [QUALIFY search_conditions]
```

Table 599 Parameter Description

Parameter Name	explain
col_ name	Specify the data columns for grouping, with multiple columns separated by ','. col_ Name can be an alias defined using AS in SELECT.
expr	Specify the expression for grouping, with multiple columns

Parameter Name	explain
	separated by ','. Attention: The col above_ The data columns or expressions defined in name and expr can not be data columns between "SELECT col_name_1,..., col_name_n from", which is a special syntax in GBase 8a MPP Cluster.
position	The ordinal of "col_name_1,..., col_name_n" between "SELECT col_name_1,..., col_name_n From", where position is an integer value that starts from 1. For example, in the "SELECT stu no, stu name from student GROUP BY 1;" statement, "1" refers to the data column stu_ no。
HAVING where_ definition	Use the GROUP BY clause to group data. For the groups formed by the GROUP BY clause, use an aggregation function to calculate the values of each group. Finally, use the HAVING clause to filter out groups that do not meet the conditions.
QUALIFY search_ conditions	The QUALIFY clause is a potential additional filter that specializes in conditional filtering and data filtering for analysis functions. If there is a GROUP BY clause, the columns in the QUALIFY clause must be consistent with the restrictions in the HAVING clause. For example, it needs to comply with the strict GROUP BY clause. If the column in the QUALIFY clause is not in the GROUP BY clause, syntax error will be reported. Single columns include columns in partition by and order by.

be careful

Each element in the HAVING clause does not need to appear in the SELECT list.

The HAVING clause restricts grouping, not rows, so aggregate function can be used.

Search in the QUALIFY clause_ The condition does not support the existence of columns of blob and long blob types.

Search in the QUALIFY clause_ Condition does not support conditional join subqueries.

Queries with a QUALIFY clause must have an analysis function, which can appear in projection columns or in the QUALIFY clause.

The analysis function does not support existence in in/not in sub queries.

Aliases are not supported in in/not in sub queries.

Example

Example 1: Group by gender and calculate the number of people of different genders.

```
gbase> SELECT CASE stu_ Sex When 0 THEN 'male' ELSE 'female' END AS stu_ sexname,COUNT(stu_sex) AS stu_ count FROM student GROUP BY stu_ sex;
+-------------+-----------+
| stu_ sexname | stu_ count |
+-------------+-----------+
|Male | 4|
|Female | 6|
+-------------+-----------+
2 rows in set
```

Example 2: Group by serial number to find students with a total score of greater than 180 in English and mathematics.

View students' math, foreign language, and overall grades in these two subjects:

```
gbase> SELECT a.stu_ name,math,english,sum(math+english) AS total FROM student a INNER JOIN result b ON a.stu_ no = b.stu_ no GROUP BY a.stu_ no ORDER BY a.stu_ no;
+----------+------+---------+-------+
| stu_ name | math | english | total |
+----------+------+---------+-------+
| Tom          | 80.0 |       85.2 | 165.2 |
| Jim          | 78.0 |       95.5 | 173.5 |
```

```
| John        | 89.5 |       99.0 | 188.5 |
| Rose        | 65.0 |       75.5 | 140.5 |
| Jane        | 92.0 |       94.0 | 186.5 |
| Mike        | 72.5 |       86.0 | 158.5 |
| Jack        | 85.0 |       76.0 | 161.5 |
| Jerry       | 95.0 |       97.0 | 192.0 |
| Allen       | 56.0 |       78.0 | 134.0 |
| Max         | 86.0 |       93.0 | 179.0 |
+----------+------+---------+-------+
10 rows in set
```

View students with a total score of over 180 in mathematics and English, and sort them in descending order of total score:

```
gbase> SELECT a.stu_ name,math,english,SUM(math+english) AS total FROM student a INNER JOIN result b ON a.stu_ no = b.stu_ no GROUP BY a.stu_ no HAVING total > 180 ORDER BY total DESC;
+----------+------+---------+-------+
| stu_ name | math | english | total |
+----------+------+---------+-------+
| Jerry       | 95.0 |       97.0 | 192.0 |
| John        | 89.5 |       99.0 | 188.5 |
| Jane        | 92.0 |       94.0 | 186.5 |
+----------+------+---------+-------+
3 rows in set
```

Example 3: View the first row of information in the entire table, grouped by gender and sorted in descending order of scores.

```
select * from t;
+------+------+-------+-------+
| id     | sex    | name    | score |
+------+------+-------+-------+
|6 | Male | name1 | 99|
|3 | Male | name3 | 68|
|5 | Female | name4 | 98|
|4 | Female | name1 | 100|
|3 | Male | name3 | 78|
|4 | Female | name4 | 97|
+------+------+-------+-------+
6 rows in set (Elapsed: 00:00:00.02)

SELECT * FROM t QUALIFY ROW_ NUMBER() OVER(PARTITION BY sex ORDER BY score)=1;
+------+------+-------+-------+
```

```
| id     | sex    | name    | score |
+------+------+-------+-------+
|4 | Female | name4 | 97|
|3 | Male | name3 | 68|
+------+------+-------+-------+
2 rows in set (Elapsed: 00:00:00.11)

SELECT *, ROW_ NUMBER() OVER(PARTITION BY sex ORDER BY score) AS row FROM t QUALIFY row=1;
+------+------+-------+-------+-----+
| id     | sex    | name    | score | row |
+------+------+-------+-------+-----+
|4 | Female | name4 | 97 | 1|
|3 | Male | name3 | 68 | 1|
+------+------+-------+-------+-----+
2 rows in set (Elapsed: 00:00:00.11)
```

Example 4: View the first row of information in the entire table after grouping by gender and jumping by score.

```
SELECT *,RANK() OVER(PARTITION BY sex ORDER BY score) AS rank FROM t QUALIFY rank=1;
+------+------+-------+-------+------+
| id     | sex    | name    | score | rank |
+------+------+-------+-------+------+
|4 | Female | name4 | 97 | 1|
|3 | Male | name3 | 68 | 1|
+------+------+-------+-------+------+
2 rows in set (Elapsed: 00:00:00.10)

SELECT * FROM t QUALIFY RANK() OVER(PARTITION BY sex ORDER BY score)=1;

+------+------+-------+-------+
| id     | sex    | name    | score |
+------+------+-------+-------+
|4 | Female | name4 | 97|
|3 | Male | name3 | 68|
+------+------+-------+-------+
2 rows in set (Elapsed: 00:00:00.11)
```

5.1.10.1.3 ORDER BY...

Function Description

ORDER BY is used to sort the result set, which is sorted based on data column names, expressions, or constants.

Grammar format

```
ORDER BY {col_name | expr | position} [ASC | DESC] , ...
```

Table 5-100 Parameter Description

Parameter Name	explain
col_ name	Specify the data columns for sorting, with multiple columns separated by ','. col_ Name can be an alias defined using AS in SELECT.
expr	Specify the expression for sorting, with multiple columns separated by ','.
position	The ordinal of "col_name_1,..., col_name_n" between "SELECT col_name_1,..., col_name_n From", where position is an integer value that starts with "1". For example, in the statement 'SELECT stu_no, stuname from student ORDER BY 1;', '1' refers to the data column stu_ no。
ASC | DESC	If you want to sort records, you can use the ASC or DESC keywords to specify the sorting rule, where ASC represents the ascending rule and DESC represents the descending rule. By default, records are sorted in ascending order.

Example

Example 1: ORDER BY...。

```
gbase> SELECT a.stu_ name,math,english,sum(math+english) AS total FROM student a INNER JOIN result b ON a.stu_ no = b.stu_ no GROUP BY a.stu_ no ORDER BY a.stu_ no;
+----------+------+---------+-------+
| stu_ name | math | english | total |
+----------+------+---------+-------+
| Tom      | 80.0 |    85.2 | 165.2 |
| Jim      | 78.0 |    95.5 | 173.5 |
```

```
| John        | 89.5 |       99.0 | 188.5 |
| Rose        | 65.0 |       75.5 | 140.5 |
| Jane        | 92.0 |       94.0 | 186.5 |
| Mike        | 72.5 |       86.0 | 158.5 |
| Jack        | 85.0 |       76.0 | 161.5 |
| Jerry       | 95.0 |       97.0 | 192.0 |
| Allen       | 56.0 |       78.0 | 134.0 |
| Max         | 86.0 |       93.0 | 179.0 |
+----------+------+----------+-------+
10 rows in set
```

5.1.10.1.4LIMIT...

Function Description

Return the result set according to the execution rules.

Grammar format

```
LIMIT {[offset,] row_count | row_count OFFSET offset}
```

Table 5101 Parameter Description

Parameter Name	explain
offset	Specify the offset of the result set, with an initial offset of 0 (instead of 1), which corresponds to the first row of the result set returned by SELECT.
row_ count	Specify the number of rows to return the result set, which is an integer value. If row_ If the value specified by count is greater than the result set after SELECT, then row_ Count will not work.

explain

- LIMIT row_ count
- Equivalent to
- LIMIT 0, row_ count
- Or equivalent to
- LIMIT row_ count OFFSET 0

Example

Example 1: Returns a result set of 10 rows

```
gbase> SELECT SUM(lo_quantity),lo_ orderkey FROM ssbm.lineorder
GROUP BY lo_ orderkey ORDER BY lo_ orderkey  LIMIT 10;
+------------------+-------------+
| SUM(lo_quantity) | lo_ orderkey |
+------------------+-------------+
|               61 |           1 |
|              149 |           2 |
|              151 |           3 |
|               30 |           4 |
|               41 |           5 |
|              191 |           6 |
|               12 |           7 |
|               66 |          32 |
|              184 |          33 |
|               75 |          34 |
+------------------+-------------+
10 rows in set
```

Example 2: Table t1 contains 10 rows of data, using the form of LIMITED m OFFSET n to display the results after executing the SELECT statement. View all 10 row result sets.

```
gbase> SELECT * FROM t1 LIMIT 10 Offset 0;
+------+
| a    |
+------+
|    1 |
|    2 |
|    3 |
|    4 |
|    5 |
|    6 |
|    7 |
|    8 |
|    9 |
|   10 |
+------+
10 rows in set
```

Example 3: Starting from the position with an offset of 2 in the result set, a 3-row result set is returned. Since the offset value of the first row of the SELECT result set is 0, the third row of the SELECT result set is the starting position with an offset of 2, and 3 rows of the result set are taken from here.

```
gbase> SELECT * FROM t1 LIMIT 3 OFFSET 2;
+------+
```

```
| a      |
+------+
|      3 |
|      4 |
|      5 |
+------+
3 rows in set

gbase> SELECT * FROM t1 LIMIT 2,3;
+------+
| a      |
+------+
|      3 |
|      4 |
|      5 |
+------+
3 rows in set
```

5.1.10.1.5Hierarchical Query

Function Description

The data in a tree structure is stored in a table, and the hierarchical relationship between the data, namely the parent-child relationship, is described by the relationship between the columns in the table. By determining the parent node of each node, the structure of the entire tree can be determined. Using START With in the SELECT command CONNECT BY can query the tree structure relationship in the table. Hierarchical queries are recursively executed through CONNECT BY. If a cycle is detected, GBase 8a MPP Cluster will default to reporting an error and exiting; If the user specifies NOCYCLE, GBase 8a MPP Cluster will return the queried records before the occurrence of the cycle.

Grammar format

START WITH... CONNECT BY syntax form:

```
SELECT column_ list|[LEVEL]
FROM [[vc_name.]database_ name.] single_ table
[WHERE …]
[hierarchical_clause]
[GROUP BY …]
[ORDER [SIBLINGS] BY …]
hierarchical_ clause :
   [START WITH <conditions>] CONNECT BY <connect_ conditions>
   | CONNECT BY <connect_ conditions> [START WITH <conditions>]
```

```
    [ORDER SIBLINGS BY {col_name | expr | position} [ASC | DESC] , ...]
connect_ condition:
   PRIOR expr1 op expr2
   | expr1 op PRIOR expr2
   | expr op connect_ condition
   | expr
```

Table -5102 Parameter Description

Parameter Name	explain
LEVEL	Pseudo column, automatically maintained by GBase 8a MPP Cluster; Used to identify the hierarchy of hierarchical query results, starting from 1.
START WITH <*conditions*>	The condition after START With identifies all root rows of the hierarchical query, and the START With clause can be omitted.
CONNECT BY <*connect_ conditions*>	The condition after CONNECT BY identifies the connection condition between parent row and child row; The expression in condition needs to specify through PRIOR whether the column involved in the expression belongs to parent row or child row. For example: ... PRIOR expr_ left = expr_ The columns involved in the right//left expression are from parent row; ... expr_ left = PRIOR expr_ The columns involved in the right//right expression are from the parent row;
PRIOR	The unary operator is only used for the condition after CONNECT BY to identify the column involved in the following expression from parent row. PRIOR only takes effect for the expression immediately following, as shown in the following example: expr_ 1 = expr_ 2 AND expr_ 3 = PRIOR expr_ 4//PRIOR only applies to expr_ 4 Effective. PRIOR cannot be nested for use, as an error will be reported in the following

Parameter Name	explain
	example: PRIOR (expr_1 + PRIOR expr2)
CONNECT_ BY_ ROOT	A unary operator used to obtain a column value corresponding to the root record of each record in the result set. Parameters can specify multiple columns, expressions, or functions. Cannot appear in connect_ Condition and Join_ In the condition.
connect_ by_ isleaf	Pseudo column, automatically maintained by GBase 8a; Used to indicate that the current layer is already the last layer. 0 represents non last layer; 1 represents the last layer. Cannot appear in connect_ Condition and Join_ In the conditions.
connect_ by_ iscycle	Pseudo column, automatically maintained by GBase 8a. Used to indicate whether a cycle has occurred in the current layer. 0 indicates that no cycle has occurred; 1 indicates that a cycle has occurred. Only NOCYCLE can be used, otherwise an error will be reported.
SYS_ CONNECT_ BY_ PATH(column,char)	The function can output a column of hierarchical query results according to a hierarchical path by using specified characters as connectors. Cannot appear in connect_ Condition and Join_ In the conditions.
Where	All connection conditions in the Where clause are in hierarchical_ Execute before clause, all filtering conditions are in hierarchical_ Execute after clause

explain

- The hierarchical query clauses' connect by 'and' start with 'do not allow columns from outer tables to appear;
- The hierarchical query from clause must be a table and must be a replicated table;
- Hierarchical queries can appear as sub queries, but sub queries are not allowed in hierarchical queries;
- Connect by association conditions cannot contain or operations, and must include equivalent conditions between parent and child nodes. The two sides of the equal sign must have different dimensions (one containing prior and the other not);
- Prior is a unary operator with the same priority as the sign and can only be used in the connect by clause; Prior cannot be followed by pseudo columns (level, rowid, etc.), expressions containing pseudo columns, nested prior, or aggregate function;
- Order siblings by can only be used in hierarchical query statements and cannot coexist with aggregates, OLAP functions, or ORDER BY;
- Real time loop detection is not supported, and can only ensure that the loop can be detected in the end. If the data volume is too large (such as over 100000), it will take a long time to detect;
- The maximum number of nodes is MAX_ INT (2147483647) (number of all nodes);
- Add three new reserved word: start, level, and prior;
- The condition after CONNECT BY is not allowed to contain subquery;

Example

Example 1: Level is used in the CONNECT BY position.

```
USE test;
DROP TABLE IF EXISTS t1;
CREATE TABLE t1(a int, b int, c char(10), d varchar(20), e varchar(5), f datetime, g decimal(6,2)) REPLICATED;
INSERT INTO t1 VALUES(0,1,'DMD','kds','dmd','2013-4-1 10:23:01',1.1);
INSERT INTO t1 VALUES(0,3,'DMD','cj','dmd','2013-4-1 10:23:01',2.1);
INSERT INTO t1 VALUES (1,3,'DMD','lm','dmd1','2013-4-1 10:23:01',2.2);
INSERT INTO t1 VALUES (1,4,'DMD','zx','dmd2','2013-4-1 10:23:01',2.3);
DROP TABLE IF EXISTS t2;
CREATE TABLE t2 REPLICATED AS SELECT * FROM t1;
gbase> SELECT level, t1.* FROM t1 CONNECT BY NOCYCLE PRIOR b = level START WITH a = 0;
+-------+------+------+------------+------+------+---------------------+------+
| level | a    | b    | c          | d    | e    | f                   | g
```

```
|
+-------+------+------+------------+------+------+---------------------+------+
|         1 |        0 |        1 | DMD               | kds    | dmd    | 2013-04-01 10:23:01 |
1.10 |
|         1 |        0 |        2 | DMD               | cj     | dmd    | 2013-04-01 10:23:01 | 2.10
|
|         2 |        0 |        1 | DMD               | kds    | dmd    | 2013-04-01 10:23:01 |
1.10 |
|         2 |        1 |        3 | DMD               | lm     | dmd1 | 2013-04-01 10:23:01 |
2.20 |
|         2 |        1 |        4 | DMD               | zx     | dmd2 | 2013-04-01 10:23:01 | 2.30
|
+-------+------+------+------------+------+------+---------------------+------+
5 rows in set (Elapsed: 00:00:10.07)
```

Example 2:

```
gbase> create table dep(depid int,depname varchar(100),upperdepid int)
replicated;
Insert into dep values (0, 'General Manager's Office', NULL);
Insert into dep values (1, 'Development Department', 0);
Insert into dep values (2, 'Testing Department', 0);
Insert into dep values (3, 'Server Development Department', 1);
Insert into dep values (3, 'Client Development Department', 1);
Insert into dep values (5, 'TA testing department', 2);
Insert into dep values (6, 'Project Testing Department', 2);
gbase> select * from dep;
+-------+-----------------+------------+
| depid | depname                 | upperdepid |
+-------+-----------------+------------+
|0 | General Manager's Office | NULL|
|1 | Development Department | 0|
|2 | Testing Department | 0|
|3 | Server Development Department | 1|
|4 | Client Development Department | 1|
|5 | TA Testing Department | 2|
|6 | Project Testing Department | 2|
+-------+-----------------+------------+
select depname , connect_ by_ root depname    "root ", connect_ by_ isleaf
"isleaf " , level ,
sys_ connect_ by_ path(depname,'/')   "path"   from dep
start with upperdepid is null connect by prior depid=upperdepid;
+-----------------+-----------+--------+-------+-----------------
|    depname    |    root    |    isleaf   |  level  |    path          |
+-----------------+-----------+--------+-------+-----------------
```

```
|    总经办      |   总经办    |    0        |      1      | /总经办                |
|    开发部      |   总经办    |    0        |      2      | /总经办/开发部         |
|Server Development Department | General Manager's Office | 1 | 3 |/General Manager's Office/Development Department/Server Development Department|
|Client Development Department | General Manager's Office | 1 | 3 |/General Manager's Office/Development Department/Client Development Department|
|    测试部      | 总经办      |  0 |2 | /总经办/测试部                        |
|TA Testing Department | General Manager's Office | 1 | 3 |/General Manager's Office/Testing Department/TA Testing Department|
|Project Testing Department | General Manager's Office | 1 | 3 |/General Manager's Office/Testing Department/Project Testing Department|
+----------------+-----------+--------+-------+-----------------
```

5.1.10.1.6Partition table Specify Partition Name Query

Grammar format

```
SELECT ... FROM [vc_name.][database_name.]<table_ name> [PARTITION (partition_name1[, partition_name2, ...])]
```

Table -5103 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table Name
partition_ name	Partition name.

explain

- When querying, one partition name or multiple partition names can be specified;
- If it is a sub partition table, you can specify the partition name, sub partition name, or any combination of partition name and sub partition name.
- An error occurred when specifying a partition that does not exist;

Example

Example 1: Without sub partitions

```
gbase> CREATE TABLE t1(id INT,vc VARCHAR(20)) PARTITION BY
LIST(ID) PARTITIONS 3 (PARTITION    p0 VALUES IN (1),PARTITION
p1 VALUES IN (2,3),PARTITION    p2 VALUES IN (4,6));
Query OK, 0 rows affected (Elapsed: 00:00:00.14)

gbase> INSERT INTO t1 VALUES(1,'a'),(2,'b'),(3,'c'),(4,'d'),(6,'e');
Query OK, 5 rows affected (Elapsed: 00:00:00.12)
Records: 5    Duplicates: 0    Warnings: 0

gbase> SELECT * FROM t1 PARTITION(p0,p2);
+------+------+
| id     | vc     |
+------+------+
|       1 | a       |
|       4 | d       |
|       6 | e       |
+------+------+
3 rows in set (Elapsed: 00:00:00.04)
```

Example 2: With sub partitions

```
gbase> CREATE TABLE t1 (id INT)
                 PARTITION BY RANGE(id)
                 SUBPARTITION BY HASH(ID)
               (
                    PARTITION    p0 VALUES LESS THAN (100)
                     (
               SUBPARTITION p0_ sp0,
               SUBPARTITION p0_ sp1
               )
               ,
               PARTITION p1 VALUES LESS THAN (200)
               (
                    SUBPARTITION p1_ sp0,
                    SUBPARTITION p1_ sp1
               )
```

```
                ) ;
Query OK, 0 rows affected (Elapsed: 00:00:00.14)

gbase> INSERT INTO t1 VALUES (1),(2),(3),(4),(5);
Query OK, 5 rows affected (Elapsed: 00:00:00.10)
Records: 5  Duplicates: 0  Warnings: 0

gbase>  INSERT INTO t1 VALUES (101),(102),(103),(104),(105);
Query OK, 5 rows affected (Elapsed: 00:00:00.10)
Records: 5  Duplicates: 0  Warnings: 0

gbase> SELECT *  FROM t1 PARTITION(p0,p1_sp0);
+------+
| id   |
+------+
|    2 |
|    4 |
|    1 |
|    3 |
|    5 |
|  102 |
|  104 |
+------+
7 rows in set (Elapsed: 00:00:00.04)
```

5.1.10.2 UNION

Function Description

UNION is used to merge the results of multiple SELECT statements into a single result set.

The selection column at the corresponding position of each SELECT statement should have the same type. (For example, the first column selected by the first statement should have the same type as the first column selected by other statements). The column name used in the first SELECT statement is used as the column name that returns the result.

The SELECT statement is a general query statement, but with the following constraints:

- Different data types cannot use UNION and UNION ALL, such as numerical,

character, date, and time types. However, DATETIME and TIMESTAMP types can use UNION and UNION ALL, while other date and time types cannot.

For example, the error reported by the union of long and varchar types is as follows:

```
gbase> SELECT c_ custkey FROM ssbm.customer UNION SELECT c_ name FROM ssbm.customer LIMIT 1;
ERROR 1733 (HY000): Gbase general error: UNION/INTERSECT/MINUS of non-matching columns: LONGLONG UNION/INTERSECT/MINUS VARCHAR
```

- If the data types on both sides of UNION and UNION ALL are CHAR, when performing UNION and UNION ALL operations, the result returns VARCHAR type.

  For example, the result set of SELECT CHAR (10) UNION SELECT CHAR (255) is VARCHAR (255).

- UNION and UNION ALL results are converted from small data types to large data types, such as INT ->BIGINT ->DECIMAL ->DOUBLE.

  For example, the result set of SELECT INT UNION SELECT DOUBLE is of DOUBLE type.

- NULL can be used as UNION and UNION ALL with any type.

  For example: SELECT NULL UNION SELECT 1;

- If only UNION is used, a result set without duplicate records will be returned, in which case UNION is equivalent to UNION DISTINCT. If UNION ALL is used, all selected result sets will be returned, and there are duplicate records in this result set.

- If both UNION [DISTINCT] and UNION ALL exist in multiple SELECT UNION queries, UNION ALL will be ignored and the result set after UNION [DISTINCT] will be returned (filtering out duplicate record rows).

- If you want to use an ORDER BY or LIMITED clause to classify or limit the entire UNION result, you can add parentheses to individual SELECT statements or place ORDER BY or LIMITED last.

Grammar format

```
select_ statement1
UNION [ALL | DISTINCT]
select_ statement2
[UNION [ALL | DISTINCT]
select_ statement3
……
UNION [ALL | DISTINCT]
select_ statementN
]
```

Table -5104 Parameter Description

Parameter Name	explain
select_ statement	SELECT statement.

Example

Example 1: Using the ORDER BY clause in a SELECT statement to classify UNION results. This type of ORDER BY cannot use column references that include table names (such as names in the form of table_name.com). Relatively, a column alias can be provided in the first SELECT statement and referenced in the ORDER BY.

```
gbase> CREATE TABLE student (stu_no int, stu_name varchar(200),stu_ sex int);
Query OK, 0 rows affected

gbase> INSERT INTO student VALUES(4,'King',1),(5,'Smith',1);
Query OK, 2 rows affected
Records: 2    Duplicates: 0    Warnings: 0

gbase> SELECT stu_ name FROM student WHERE stu_ name LIKE 'S%'
       UNION
       SELECT stu_ name FROM student WHERE stu_ name LIKE '%K%'
       ORDER BY stu_ name LIMIT 10;
+----------+
| stu_ name |
+----------+
| King         |
| Smith       |
+----------+
2 rows in set
```

Example 2: When applying ORDER BY or LIMITED to a separate SELECT, insert a clause into the parentheses that encapsulate the SELECT. The ORDER BY of the SELECT statement in parentheses only works when combined with LIMITED. Otherwise, ORDER BY will be optimized to prevent syntactic ambiguity.

```
gbase> (SELECT c_name FROM ssbm.customer WHERE c_name LIKE '%00002%' ORDER BY c_name LIMIT 10)
       UNION
       (SELECT c_name FROM ssbm.customer WHERE c_name LIKE '%00003%'ORDER BY c_name LIMIT 10);
+--------------------+
| c_ name                    |
```

```
+--------------------+
| Customer#000000002 |
| Customer#000000020 |
| Customer#000000021 |
| Customer#000000022 |
| Customer#000000023 |
| Customer#000000024 |
| Customer#000000025 |
| Customer#000000026 |
| Customer#000000027 |
| Customer#000000028 |
| Customer#000000003 |
| Customer#000000030 |
| Customer#000000031 |
| Customer#000000032 |
| Customer#000000033 |
| Customer#000000034 |
| Customer#000000035 |
| Customer#000000036 |
| Customer#000000037 |
| Customer#000000038 |
+--------------------+
20 rows in set
```

Example 3: The length and type of columns in the UNION result set need to take into account the values found from all SELECT statements. The value found by the first SELECT is shorter than the value found by the second SELECT.

```
gbase> SELECT REPEAT('a',1) UNION SELECT REPEAT('b',10);
+---------------+
| REPEAT('a',1) |
+---------------+
| a             |
| bbbbbbbbbb    |
+---------------+
2 rows in set
```

Example 4: UNION of DATETIME and TIMESTAMP types.

```
gbase> CREATE TABLE t_ student (sno int, sname varchar(100),sdate datetime);
Query OK, 0 rows affected

gbase> INSERT INTO t_ student VALUES(1,'Tom',NOW()),(2,'Jim',NOW());
Query OK, 2 rows affected
```

```
Records: 2  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t_ student;
+------+-------+---------------------+
| sno  | sname | sdate               |
+------+-------+---------------------+
|    1 | Tom   | 2013-10-14 17:56:19 |
|    2 | Jim   | 2013-10-14 17:56:19 |
+------+-------+---------------------+
2 rows in set

gbase> CREATE TABLE t_ Result (sno int, sresult decimal(10,2),sdate timestamp);
Query OK, 0 rows affected

gbase> INSERT INTO t_ result VALUES(1,99.5,NOW()),(2,100,NOW());
Query OK, 2 rows affected
Records: 2  Duplicates: 0  Warnings: 0

gbase> SELECT * FROM t_ result;
+------+---------+---------------------+
| sno  | sresult | sdate               |
+------+---------+---------------------+
|    1 |   99.50 | 2013-10-14 17:57:15 |
|    2 |  100.00 | 2013-10-14 17:57:15 |
+------+---------+---------------------+
2 rows in set

gbase> SELECT sno,sdate FROM t_ student UNION SELECT sno,sdate FROM t_ result;
+------+---------------------+
| sno  | sdate               |
+------+---------------------+
|    1 | 2013-10-14 17:56:19 |
|    2 | 2013-10-14 17:56:19 |
|    1 | 2013-10-14 17:57:15 |
|    2 | 2013-10-14 17:57:15 |
+------+---------------------+
4 rows in set
```

Example 5: CHAR and UNION of CHAR types.

```
gbase> USE test;
Query OK, 0 rows affected
```

```
gbase> CREATE TABLE t1(a char(10));
Query OK, 0 rows affected

gbase> CREATE TABLE t2(a char(255));
Query OK, 0 rows affected

gbase> CREATE TABLE ta AS SELECT * FROM t1 UNION SELECT *
FROM t2;
Query OK, 0 rows affected

gbase> DESC ta;
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| a     | varchar(255) | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
1 row in set
```

5.1.10.3 INTERSECT

Function Description

INTERSECT (intersection operator) returns the same result set in each SELECT query result, which means that the common parts of multiple query result sets are used as the final returned result set. Also, do not ignore null values for transportation calculations.

Grammar format

```
select_ statement1
INTERSECT
select_ statement2
```

Table 5105 Parameter Description

Parameter Name	explain
select_ statement	SELECT statement.

Example

Example 1: SELECT INTERSECT SELECT ...

Tables and data used in the example:

```
CREATE TABLE t1 (a int , b varchar(10));
CREATE TABLE t2 (c int ,d varchar(20),e varchar(5));
INSERT INTO t1 VALUES(1,'a'),(2,'b'),(3,'c');
```

```
INSERT INTO t2 VALUES(1,'a','aa'),(2,'b','bb'),(4,'c','cc');
```

INTERSECT execution result:

```
gbase> SELECT a ,b FROM t1;
+------+------+
| a    | b    |
+------+------+
|    1 | a    |
|    2 | b    |
|    3 | c    |
+------+------+
3 rows in set

gbase> SELECT c AS a, d AS b FROM t2;
+------+------+
| a    | b    |
+------+------+
|    1 | a    |
|    2 | b    |
|    4 | c    |
+------+------+
3 rows in set

gbase> SELECT a ,b FROM t1 INTERSECT SELECT c AS a, d AS b FROM
t2;
+------+------+
| a    | b    |
+------+------+
|    1 | a    |
|    2 | b    |
+------+------+
2 rows in set
```

5.1.10.4 MINUS

Function Description

The MINUS (difference operator) returns the result set of the first SELECT statement, and the information contained in the query results of this result set cannot appear in the result set of the second query statement. Additionally, the difference operation does not ignore null values.

Grammar format

```
select_ statement1
```

```
MINUS
select_ statement2
```

Table -5106 Parameter Description

Parameter Name	explain
select_ statement	SELECT statement.

Example

Example 1: SELECT MINUS SELECT...

Tables and data used in the example:

```
CREATE TABLE t1 (a int , b varchar(10));
INSERT INTO t1 VALUES(1,'a'),(2,'b'),(3,'c');
INSERT INTO t1 VALUES(null,null);
CREATE TABLE t2 (c int ,d varchar(20),e varchar(5));
INSERT INTO t2 VALUES(1,'a','aa'),(2,'b','bb'),(4,'c','cc');
```

Minus execution result:

```
gbase> SELECT a,b FROM t1;
+------+------+
| a    | b    |
+------+------+
|    1 | a    |
|    2 | b    |
|    3 | c    |
| NULL | NULL |
+------+------+
4 rows in set

gbase> SELECT c AS a, d AS b FROM t2;
+------+------+
| a    | b    |
+------+------+
|    1 | a    |
|    2 | b    |
|    4 | c    |
+------+------+
3 rows in set

gbase> SELECT a ,b FROM t1 MINUS SELECT c AS a, d AS b FROM t2;
+------+------+
| a    | b    |
+------+------+
|    3 | c    |
```

```
| NULL | NULL |
+------+------+
2 rows in set
```

5.1.10.5 EXCEPT

Function Description

The usage of EXCEPT is consistent with the Minus (difference operator), and the returned result set is the result set of the first SELECT statement, and the information contained in the query results of this result set cannot appear in the result set of the second query statement. Additionally, this operation does not ignore null values.

Grammar format

```
select_ statement1
EXCEPT
select_ statement2 ;
```

Table 5107 Parameter Description

Parameter Name	explain
select_ statement	SELECT statement.

Example

Example 1: SELECT EXCEPT SELECT...

Tables and data used in the example:

```
CREATE TABLE t1 (a int , b varchar(10));
INSERT INTO t1 VALUES(1,'a'),(2,'b'),(3,'c');
INSERT INTO t1 VALUES(null,null);
CREATE TABLE t2 (c int ,d varchar(20),e varchar(5));
INSERT INTO t2 VALUES(1,'a','aa'),(2,'b','bb'),(4,'c','cc');
```

Minus execution result:

```
gbase> SELECT a,b FROM t1;
+------+------+
| a    | b    |
+------+------+
|    1 | a    |
|    2 | b    |
|    3 | c    |
| NULL | NULL |
+------+------+
4 rows in set
```

```
gbase> SELECT c AS a, d AS b FROM t2;
+------+------+
| a    | b    |
+------+------+
|    1 | a    |
|    2 | b    |
|    4 | c    |
+------+------+
3 rows in set

gbase> SELECT a ,b FROM t1 EXCEPT SELECT c AS a, d AS b FROM t2;
+------+------+
| a    | b    |
+------+------+
|    3 | c    |
| NULL | NULL |
+------+------+
2 rows in set
```

5.1.10.6 CTE(with…as…)

Function Description

CTE (common table expression) can define multiple, in writing order, and supports subsequent CTE queries_ The CTE defined earlier is referenced in the definition. Expression_ The name can be the same as the base table or view name in the database, and the main query references expression_ The name identifier refers to CTE instead of the underlying table or view in the database. If the library name is not written, the defined CTE will be referenced first. If the library name and table name are written, such as test.t1, the underlying table or view will be referenced.

Grammar format

```
[
  WITH
  Expression_ name AS
   (
      CTE_ query_ definition
   )
   , …
]
SELECT... (main query);
```

- CTE_ query_ Definition is a select statement that only supports select query statements and not other statements. Its syntax must comply with the syntax of

GCluster's select query. Other syntax constraints are consistent with the from subquery (such as defining statements that cannot be select... into outfile, select... into... server, etc.). The syntax supported in From subqueries and CTE definition statements are also supported.

- Expression_ Name is the given CTE name, which must be an identifier that complies with the GCluster naming convention. The CTE name is limited to 64 characters in length and consists of numbers, letters, and underscores. If it contains special characters, it must be enclosed in back quotes. If multiple CTEs are defined, the name of each CTE must be unique.

Example

```
gbase> set _ t_ gcluster_ support_ cte=1;
Example 1: Defining a CTE
gbase> with tt as (select * from t) select * from tt;
+------+------+
| a    | b    |
+------+------+
|    1 | aa   |
+------+------+
1 row in set (Elapsed: 00:00:00.76)
Example 2: Defining Multiple CTEs
gbase> with tt1 as (select a,b from t),tt2 as (select a,b from t) select * from tt1
join tt2 on tt1.a=tt2.a;
+------+------+------+------+
| a    | b    | a    | b    |
+------+------+------+------+
|    1 | aa   |    1 | aa   |
+------+------+------+------+
1 row in set (Elapsed: 00:00:04.23)
Example 3: The subsequent CTE references the previous CTE
gbase> with tt1 as(select a,b from t),tt2 as(select a+10 as ab from tt1) select *
from tt2;
+------+
| ab   |
+------+
|   11 |
+------+
1 row in set (Elapsed: 00:00:00.54)
Example 4: Define once, use multiple times
gbase> with tt as(select a,b from t) select * from tt tt1 join tt tt2 on tt1.a=tt2.a;
+------+------+------+------+
| a    | b    | a    | b    |
+------+------+------+------+
|    1 | aa   |    1 | aa   |
```

```
+------+------+------+------+
1 row in set (Elapsed: 00:00:02.86)
Example 5: Union scenario, only supports' global 'CTE
gbase> with tt as(select a,b from t) select * from tt where a>1 union select * from tt where a<1;
+------+------+
| a    | b    |
+------+------+
|    2 | bbb  |
|   -1 | ccc  |
+------+------+
4  rows in set (Elapsed: 00:00:01.01)
Example 6: Use CTE in the select section of insert... select
gbase> insert into t(a,b) with tt as (select a,b from t) select a,b from tt;
Query OK, 3 rows affected (Elapsed: 00:00:01.29)
Records: 3  Duplicates: 0  Warnings: 0
Example 7: The select section in create... select uses CTE
gbase> create table abc as with tt as(select a,b from t) select a,b from tt;
Query OK, 6 rows affected (Elapsed: 00:00:05.91)
Example 8: Define CTE in a subquery
gbase> with base as (select * from t) select * from (with t1 as (select * from base) select * from t1) t where a in (select a from t);
+------+------+
| a    | b    |
+------+------+
|    1 | aa   |
|    2 | bbb  |
|   -1 | ccc  |
|    1 | aa   |
|    2 | bbb  |
|   -1 | ccc  |
+------+------+
6 rows in set (Elapsed: 00:00:02.70)
```

be careful

- Add parameters_ t_ gcluster_ support_ Cte controls whether gcluster supports cte syntax.
  This parameter is a session level parameter, with a default value of 0 indicating that it is not supported. When the value is set to 1, CTE syntax is supported.
- The alias declared with as in a subquery is global and does not support range control of identifiers in subqueries. With as in subqueries can be sequentially referenced globally and share the same namespace. In the order of definition, the following with as definitions can refer to the previous with as definition.
  with tt as (select * from t1) select * from (with tt as (select * from t1) select * from tt limit 10) aa;

  ERROR 1066 (42000): Not unique table/alias: 'tt'
- The use of grouped hint is supported in the select statement defined by CTE. Like
  gccli - c - q

  use vc vc2023;

  use testdb;

  set _ t_ gcluster_ support_ cte=1;

  with tt as(select /*+ grouped('1') */ a,b from t1) select * from tt join t2 on tt.a=t2.a;
- Query statements containing CTE are not supported in stored procedures, functions, and view definitions.
- Nested definition CTE is not supported. For example:

  with abcd as (with tt as (select 1 from t1) select * from tt) select * from abcd;

  ERROR 1149 (42000): You have an error in your SQL syntax; check the manual that corresponds to your GBase server version for the right syntax to use
- When the following parameters are set, if the CTE contains a window function, it will pass the resolution, but the execution result is incorrect

  _ t_ gcluster_ support_ cte=1

  _ t_ gbase_ new_ window_ function_ support=1

5.1.11 GBase 8a MPP Cluster Other Statements

5.1.11.1 DESCRIBE

Function Description

DESCRIBE provides column information in a table. It is a simple form of SHOW COLUMNS, which can also display view information. By parameter gbase_ show_ ident_ case_ Sensitive can control the casing of displayed column names, which is consistent with the casing of column names in the source table structure by default. Please refer to the configuration parameters of Gnode in Chapter 7.6.3 for details.

Grammar format

{DESCRIBE | DESC} [*vc_name*.][*database_name*.]<*table_ name*> [*col_name*]

Table -5108 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table Name
col_ name	It can be a column name, a string containing a wildcard character of "%" (when using "%", this wildcard character needs to be enclosed in single quotes, such as "'id%") or "_", used to obtain output for each column with a name that matches the string. When a string contains spaces or other special characters, it needs to be surrounded by quotation marks.

Example

Example 1: Viewing customer c_ The column information of custkey.

```
gbase> DESCRIBE customer c_ custkey;
+-----------+------------+------+-----+---------+-------+
| Field     | Type       | Null | Key | Default | Extra |
+-----------+------------+------+-----+---------+-------+
| c_ custkey | bigint(20) | YES  |     | NULL    |       |
+-----------+------------+------+-----+---------+-------+
1 row in set
```

Explanation of the meaning of column information in Table -5109

Parameter Name	explain
Field	Table field name.
Type	The data type of the table field.
Null	Indicates whether NULL values can be stored, and YES indicates

Parameter Name	explain
	that they can be stored.
Key	This column is empty, GBase 8a MPP Cluster does not have a key.
Default	Represents the default value assigned to this field.
Extra	Contains all additional valid information about this field. If the column type is different from the one defined in the CREATE TABLE statement, it is important to note that the column type may change

5.1.11.2 USE

Function Description

Using the specified database as the current default database does not prevent users from accessing tables in another database. The database remains the current database until the session ends or another USE statement is issued.

Grammar format

```
USE [vc_name.]<database_ name> | <VC vc_ name>;
```

Table 5110 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.

Example

Example 1: Using the ssbm database as the default database.

```
gbase> USE vc1.ssbm;
Query OK, 0 rows affected

gbase> SELECT COUNT(*) FROM customer;
+----------+
| COUNT(*) |
+----------+
```

```
|       30000 |
+----------+
1 row in set
```

Example 2: Using the gbase database as the default database to access the customer table in the ssbm database.

```
gbase> USE vc1.gbase;
Query OK, 0 rows affected

gbase> SELECT COUNT(*) FROM ssbm.customer;
+----------+
| COUNT(*) |
+----------+
|       30000 |
+----------+
1 row in set
```

5.1.11.3 KILL

Function Description

KILL thread_ The id statement can terminate a thread. When applying Gcluster 8a MPP Cluster, each connection has its own separate thread.

Grammar format

```
KILL [CONNECTION | QUERY] thread_ id
```

Table 5111 Parameter Description

Parameter Name	explain
KILL CONNECTION	KILL Connection is the same as KILL without option modification, used to terminate the specified thread_ ID thread.
KILL QUERY	Abort the statement currently executing the connection, but do not terminate the connection itself.
thread_ id	Thread ID, the SHOW [full] PROCESS LIST statement can be used to view information about the running thread.


explain

- If you have PROCESS permission, you can view all threads.
- If there is SUPER permission, all threads and statements can be terminated. Otherwise, users can only view and terminate their own threads and statements.
- When a user executes a KILL command, the corresponding thread is marked as killed. In most cases, ending a thread may take some time because this flag is only checked during a specific period.
- In the SELECT loop, the kill flag is checked after reading some branches, and if the kill flag is set, the statement terminates.
- During the alter table, check the kill flag before reading every part of the table from the source table. If set, the statement aborts and deletes the temporary table.

5.1.11.4 SET

Grammar format

```
SET [GLOBAL | SESSION] <variable_ name> = <value>
```

Table -5112 Parameter Description

Parameter Name	explain
SESSION	Omitting the SESSION keyword means that by default, it is at the session level, meaning that the current connection setting on the node machine that executes the command in the cluster is successful, and other nodes remain unchanged.
GLOBAL	When set as this keyword, the new variable value will be used in the new connection.
variable_ name	Variable name.
value	Variable value.

Example

Example 1: The default is session level, which is only valid for the current connection on the current node machine.

```
gbase> SET AUTOCOMMIT = 1;
Query OK, 0 rows affected
```

Example 2: Using the GLOBAL keyword, set the value of 'gbase_sql_trace' to 'on'.

```
gbase> SHOW VARIABLES LIKE '%trace%';
+-----------------------------+-------+
| Variable_ name              | Value |
+-----------------------------+-------+
| _ gbase_ sql_ trace_ file_ mode | OFF   |
| auto_ trace                 | OFF   |
| gbase_ sql_ trace           | OFF   |
| gbase_ sql_ trace_ level    | 0     |
+-----------------------------+-------+
4 rows in set

gbase> SET GLOBAL gbase_ sql_ trace =on;
Query OK, 0 rows affected
gbase> SHOW VARIABLES LIKE '%gbase_ sql_ trace%';
+-----------------------------+-------+
| Variable_ name              | Value |
+-----------------------------+-------+
| _ gbase_ sql_ trace_ file_ mode | OFF   |
| gbase_ sql_ trace           | OFF   |
| gbase_ sql_ trace_ level    | 0     |
+-----------------------------+-------+
3 rows in set
gbase> QUIT
Bye

# gccli -uroot -p
Enter password:

GBase client 9.5.3.17.117651. Copyright (c) 2004-2019, GBase.  All Rights Reserved.
gbase> SHOW VARIABLES LIKE '%trace%';
+-----------------------------+-------+
| Variable_ name              | Value |
+-----------------------------+-------+
| _ gbase_ sql_ trace_ file_ mode | OFF   |
| auto_ trace                 | OFF   |
| gbase_ sql_ trace           | ON    |
| gbase_ sql_ trace_ level    | 0     |
+-----------------------------+-------+
4 rows in set
```

5.1.11.5 PAUSE

Function Description

Using PAUSE thread_ The id statement can pause a thread's SELECT operation.

Grammar format

```
PAUSE thread_ id
```

Table -5113 Parameter Description

Parameter Name	explain
thread_ id	Thread ID, the SHOW [full] PROCESS LIST statement can be used to view information about the running thread.

explain

- If you have PROCESS permission, you can view all threads.
- If you have SUPER permission, you can pause/resume the SELECT operation on all threads. Otherwise, users can only view and pause/resume their own threads.
- When the user executes a PAUSE command, the corresponding thread flag is set. In most cases, this operation may take some time because the flag is only checked during a specific period.

5.1.11.6 CONTINUE

Function Description

CONTINUE thread_ The id statement can continue a thread's SELECT operation.

Grammar format

```
CONTINUE thread_ id
```

Table -5114 Parameter Description

Parameter Name	explain
thread_ id	Thread ID, the SHOW [full] PROCESS LIST statement can be used to view information about the running thread.

 **explain**

- If you have PROCESS permission, you can view all threads.
- If you have SUPER permission, you can pause/resume the SELECT operation on all threads. Otherwise, users can only view and pause/resume their own threads.
- When the user executes a Continuue command, the corresponding thread flag is set. In most cases, this operation may take some time because the flag is only checked during a specific period.

5.1.11.7 SHOW management statement

SHOW provides information about the server's database, tables, columns, or status in various forms.

Table -5115 Parameter Description

Command	Function
SHOW [FULL] COLUMNS FROM { [vc_name.][database_name.]table_name| table_name [FROM [vc_name.]database_name]} [LIKE 'pattern']	Display information about columns in a given table.
SHOW [FULL] FIELDS FROM { [vc_name.][database_name.]table_name| table_name [FROM [vc_name.]database_name]} [LIKE 'pattern']	Display information about columns in a given table.
SHOW INDEX FROM { [vc_name.][database_name.]table_name| table_name [FROM [vc_name.]database_name]}	List the indexes of the specified tables in the selected database.
SHOW [FULL] TABLES [FROM [vc_name.]database_ name] [LIKE ‘pattern’] [where conditions]	List non temporary tables for a given database.
SHOW [FULL] TABLESPACES [FROM [vc_name.]database_ name]	Display tablespace information.
SHOW {DATABASES | SCHEMAS} [LIKE 'pattern']	Display database information.
SHOW VCS	Display VC information.
SHOW ENGINES	Display ENGINES information.
SHOW TABLE STATUS [FROM [vc_name.]database_ name] { [LIKE 'pattern'] | [where conditions] }	Display information about the current status of all tables or tables in a specified database.
SHOW FUNCTION STATUS [where conditions]	Display the status of the function that has been successfully created.

Command	Function
SHOW PROCEDURE STATUS [where conditions]	Display the status of the successfully created stored procedure.
SHOW CREATE {DATABASE | SCHEMA} [vcname.]database_ name	Display the creation statement for the given database.
SHOW [FULL] CREATE TABLE [vc_name.][database_name.]table_ name [for sync]	Display the creation statement for the given table
SHOW CREATE VIEW [vc_name.][database_name.]view_ namel	Display the creation statement for the given view.
SHOW CREATE FUNCTION [vc_name.][database_name.]func_ name	Display the creation statement for the given custom function.
SHOW CREATE PROCEDURE [vc_name.][database_name.]proc_ name	Display the creation statement for the given stored procedure.
SHOW CREATE SYNONYM [vc_name.][database_name.]syn_ name	Display the creation statement for the given private synonym.
SHOW CREATE PUBLIC SYNONYM [vc_name.]sym_ name	Display the creation statement for the given public synonym.
SHOW GRANTS FOR {[user_name] | CURRENT_USER[()]}	Display GRANT statement information for a given user.
SHOW TABLE LOCKS	Display table lock information.
SHOW DISTRIBUTION TABLES [FROM [vc_name.]database_ name] [LIKE ‘pattern’]	List table information in the specified database.
SHOW OPEN TABLES FROM [vc_name.]database_ name { [LIKE 'pattern'] | [WHERE conditions] }	List the tables opened under the specified database.
SHOW PRIORITIES [WHERE CONDITIONS]	Display priority status.
SHOW [FULL] PROCESSLIST	Display the running threads.
SHOW [GLOBAL | SESSION] STATUS { [LIKE 'pattern'] | [WHERE conditions] }	Provide status information.
SHOW [GLOBAL | SESSION] VARIABLES { [LIKE 'pattern'] | [WHERE conditions] }	Display the values of some GBase 8a MPP Cluster system variables.
SHOW ERRORS [LIMIT [offset,] row_ count]	Display error messages generated by the last statement.
SHOW COUNT(*) ERRORS	Display the number of error messages generated by the last statement
SHOW WARNINGS [LIMIT [offset,] row_ count]	Display warning and caution messages generated by the last statement.
SHOW COUNT（*）WARNINGS	Display the number of warning and

Command	Function
	attention messages generated by the last statement.
SHOW GCLUSTER ENTRY	Display the nodes with the lowest number of connections in the cluster.
SHOW [GCLUSTER] NODES	Used to obtain node information for coordinator or data clusters.

5.1.11.7.1 SHOW COLUMNS

Function Description

Display information about columns in a given table. This statement also applies in the view. Same effect as SHOW FIELDS.

Grammar format

```
SHOW [FULL] COLUMNS FROM { [vc_name.][database_name.]table_name |
table_name [FROM [vc_name.]database_name]} [LIKE 'pattern'];
```

Table 5-116 Parameter Description

Parameter Name	explain
FULL	The output generated by the keyword FULL includes the permissions that the user has for each column. FULL also displays all column annotation information.
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table name.
pattern	A string that can contain SQL '%' and '_' wildcards.

Example

Example 1: Display column information.

Tables and data used in the example:

```
USE vc1.demo;
CREATE TABLE "t1" ("a" INT(11) DEFAULT NULL,"b" VARCHAR(10)
DEFAULT NULL);
```

Using VC_ name.database_ name.table_ Name format:

```
gbase> SHOW COLUMNS FROM vc1.demo.t1;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
```

```
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | YES  |     | NULL    |       |
| b     | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (Elapsed: 00:00:00.00)

Using the From table_ name FROM vc_ name.database_ Name. Format:
gbase> SHOW COLUMNS from t1 FROM vc1.demo;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | YES  |     | NULL    |       |
| b     | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (Elapsed: 00:00:00.00)

Query using LIKE 'pattern' mode:
gbase> SHOW COLUMNS from t1 FROM vc1.demo like 'a%';
+-------+---------+------+-----+---------+-------+
| Field | Type    | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| a     | int(11) | YES  |     | NULL    |       |
+-------+---------+------+-----+---------+-------+
1 row in set (Elapsed: 00:00:00.00)
```

The DESCRIBE statement provides information similar to the SHOW COLUMNS statement. Please refer to the content in "5.1.11.1 DESCRIBE". By parameter gbase_ show_ ident_ case_ Sensitive can control the casing of displayed column names, which is consistent with the casing of column names in the source table structure by default. Please refer to the configuration parameters of Gnode in Chapter 7.6.3 for details.

5.1.11.7.2SHOW FIELDS

Function Description

Display information about columns in a given table. This statement also applies in the view. The function is the same as SHOW COLUMNS.

Grammar format

```
SHOW [FULL] FIELDS FROM { [vc_name.][database_name.]table_name | table_name [FROM [vc_name.]database_name]} [LIKE 'pattern'];
```

Table 5117 Parameter Description

Parameter Name	explain
FULL	The output generated by the keyword FULL includes the permissions that the user has for each column. FULL also displays all column annotation information.
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table Name
pattern	A string that can contain SQL '%' and '_' wildcards.

Example

Example 1: Display column information.

Tables and data used in the example:

```
USE vc1.demo;
CREATE TABLE "t1" ("a" INT(11) DEFAULT NULL,"b" VARCHAR(10)
DEFAULT NULL);
```

Using VC_ name.database_ name.table_ Name format:

```
gbase> SHOW FIELDS FROM vc1.demo.t1;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | YES  |     | NULL    |       |
| b     | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (Elapsed: 00:00:00.00)
```

Using the From table_ name FROM vc_ name.database_ Name. Format:

```
gbase> SHOW FIELDS from t1 FROM vc1.demo;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| a     | int(11)     | YES  |     | NULL    |       |
| b     | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (Elapsed: 00:00:00.00)
```

Query using LIKE 'pattern' mode:

```
gbase> SHOW FIELDS from t1 FROM vc1.demo like 'a%';
+-------+---------+------+-----+---------+-------+
| Field | Type    | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| a     | int(11) | YES  |     | NULL    |       |
```

```
+-------+---------+------+-----+---------+-------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.3SHOW INDEX

Function Description

List the indexes of the specified tables in the selected database.

Grammar format

```
SHOW INDEX FROM { [vc_name.][database_name.]table_name| table_name
[FROM [vc_name.]database_name]};
```

Table 5-118 Parameter Description

Parameter Name	explain
FULL	The output generated by the keyword FULL includes the permissions that the user has for each column. FULL also displays all column annotation information.
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table Name

Example

Example 1: List the indexes of the specified tables in the demo database.

```
Tables and data used in the example:
USE vc1.demo;
CREATE TABLE t_ index(a int, b varchar(10));
CREATE INDEX idx_ a ON t_ index(a) USING HASH GLOBAL;

Using VC_ name.database_ name.table_ Name format:
gbase> SHOW INDEX FROM vc1.demo.t_ index\G
*************************** 1. row ***************************
           Table: t_ index
   Non_ unique: 1
      Key_ name: idx_ a
Seq_ in_ index: 1
 Column_ name: a
    Collation: NULL
 Cardinality: NULL
      Sub_ part: NULL
```

```
          Packed: NULL
            Null: YES
   Index_ type: GLOBAL HASH
         Comment:
1 row in set (Elapsed: 00:00:00.00)
```

Using the From table_ name FROM vc_ name.database_ Name. Format:

```
gbase>    SHOW INDEX from t_ index FROM vc1.demo\G
*************************** 1. row ***************************
           Table: t_ index
   Non_ unique: 1
       Key_ name: idx_ a
Seq_ in_ index: 1
  Column_ name: a
     Collation: NULL
  Cardinality: NULL
       Sub_ part: NULL
          Packed: NULL
            Null: YES
   Index_ type: GLOBAL HASH
         Comment:
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.4SHOW TABLES

Function Description

List non temporary tables for a given database.

 be careful

If the user does not have permission to the table, the table will not be output when using SHOW TABLES.

Grammar format

SHOW [FULL] TABLES [FROM [*vc_name.*]*database_ name*] [WHERE *conditions*] [LIKE '*pattern*'];

Table 5119 Parameter Description

Parameter Name	explain
FULL	Output Column Table_ type。 The table displays BASE TABLE, while the view displays VIEW

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name.
pattern	A string that can contain SQL '%' and '_' wildcards.
conditions	Filter conditions.

Example

Example 1: List non temporary tables in the demo database.

Tables and data used in the example:

```
USE vc1.demo;
Create table t1(a int);
Create table d1(a int);
Create view t1_ v as select * from t1;
Create view v_ d1 as select * from d1;
```

Display non temporary tables under the vc1. demo library:

```
gbase> SHOW TABLES FROM vc1.demo;
+----------------+
| Tables_ in_ demo |
+----------------+
| d1                        |
| t1                        |
| t1_ v                     |
| v_ d1                     |
+----------------+
4 rows in set (Elapsed: 00:00:00.00)
```

Use the FULL modifier to display non temporary tables under the demo library:

```
gbase> SHOW FULL TABLES FROM vc1.demo;
+----------------+------------+
| Tables_ in_ demo | Table_ type |
+----------------+------------+
| d1                        | BASE TABLE |
| t1                        | BASE TABLE |
| t1_ v                     | VIEW             |
| v_ d1                     | VIEW             |
+----------------+------------+
4 rows in set (Elapsed: 00:00:00.00)
```

Display non temporary tables starting with t in the demo library:

```
gbase> SHOW FULL TABLES FROM demo LIKE 't%';
```

```
+---------------------+------------+
| Tables_ in_ demo (t%) | Table_ type |
+---------------------+------------+
| t1                          | BASE TABLE |
| t1_ v                        | VIEW           |
+---------------------+------------+
2 rows in set (Elapsed: 00:00:00.00)

Display the views under the demo library:
gbase> SHOW FULL TABLES FROM demo where table_ type='VIEW';
+---------------+------------+
| Tables_ in_ demo | Table_ type |
+---------------+------------+
| t1_ v                  | VIEW           |
| v_ d1                  | VIEW           |
+---------------+------------+
2 rows in set (Elapsed: 00:00:00.00)

gbase> show full tables from db1 where Table_ type = 'VIEW' like 't%';
    +---------------+------------+
    | Tables_ in_ db1 | Table_ type |
    +---------------+------------+
    | t1                    | BASE TABLE |
    +---------------+------------+
    1 row in set (Elapsed: 00:00:00.01)
```

5.1.11.7.5SHOW TABLESPACES

Function Description

Query tablespace information.

Grammar format

```
SHOW [FULL] TABLESPACES [FROM [vc_name.]database_ name] ;
```

Table -5120 Parameter Description

Parameter Name	explain
FULL	Display whether it is the default tablespace.
vc_ name	VC name, optional.
database_ name	Database name.

Example

```
gbase> SHOW FULL TABLESPACES;
+-----------------------+-----------------------+------------+
```

```
| Tablespace_ in_ test_ sdy | Tablespace_ in_ test_ sdy | Is_ default |
+-------------------------+-------------------------+------------+
| sys_ tablespace           | .                         | no          |
| tbs1                      | ../ tbs1                  | yes         |
+-------------------------+-------------------------+------------+
2 rows in set (Elapsed: 00:00:00.00)
```

5.1.11.7.6SHOW DATABASES

Function Description

List the databases on the GBase 8a MPP Cluster server host. Unless users have all SHOW DATABASES permissions, they can only see the databases they have permissions on.

Grammar format

```
SHOW {DATABASES | SCHEMAS} [LIKE 'pattern']
```

Table 5121 Parameter Description

Parameter Name	explain
pattern	A string that can contain SQL '%' and '_' wildcards.

Example

```
gbase> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_ schema |
| performance_ schema |
| gbase              |
| gctmpdb            |
| demo               |
| gclusterdb         |
+--------------------+
6 rows in set (Elapsed: 00:00:00.00)

gbase> SHOW DATABASES LIKE 'd%';
+---------------+
| Database (d%) |
+---------------+
| demo          |
+---------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.7SHOW VCS

Function Description

List the VC information on the GBase 8a MPP Cluster server host.

Grammar format

```
SHOW VCS;
```

Example

```
gbase> SHOW VCS;
+---------+------+---------+
| id      | name | default |
+---------+------+---------+
| vc00001 | vc1  | Y       |
| vc00002 | vc2  |         |
+---------+------+---------+
2 rows in set (Elapsed: 00:00:00.00)
```

5.1.11.7.8SHOW ENGINES

Function Description

List the engines information on the GBase 8a MPP Cluster server host.

Grammar format

```
SHOW ENGINES;
```

Example

```
gbase> SHOW ENGINES\G
*************************** 1. row ***************************
      Engine: MRG_ GSSYS
     Support: YES
     Comment: Collection of identical GsSYS tables
Transactions: NO
          XA: NO
  Savepoints: NO
*************************** 2. row ***************************
```

```
        Engine: CSV
       Support: YES
       Comment: CSV storage engine
Transactions: NO
            XA: NO
  Savepoints: NO
*************************** 3. row ***************************
        Engine: EXPRESS
       Support: DEFAULT
       Comment: Express storage engine
Transactions: YES
            XA: YES
  Savepoints: NO
*************************** 4. row ***************************
        Engine: GsSYS
       Support: YES
       Comment: GsSYS engine
Transactions: NO
            XA: NO
  Savepoints: NO
*************************** 5. row ***************************
        Engine: MEMORY
       Support: YES
       Comment: Hash based, stored in memory, useful for temporary tables
Transactions: NO
            XA: NO
  Savepoints: NO
5 rows in set (Elapsed: 00:00:00.00)
```

5.1.11.7.9SHOW TABLE STATUS

Function Description

List information about the current state of all tables or tables in a specified database.

Grammar format

```
SHOW TABLE STATUS [FROM [vc_name.]database_ name] { [LIKE 'pattern']
| [where conditions] }
```

Table 5122 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.

Parameter Name	explain
database_ name	Database name.
pattern	A string that can contain SQL '%' and '_' wildcards.
where conditions	Filter conditions.

Example

Example 1: View the status information of a specified table in a specified database.

```
gbase> SHOW TABLE STATUS FROM vc1.demo WHERE name='t1'\G

*************************** 1. row ***************************
                           Name: t1
                         Engine: EXPRESS
                        Version: 10
                     Row_ format: Compressed
                           Rows: 0
                 Avg_ row_ length: 0
                    Data_ length: 0
                Max_ data_ length: 0
                   Index_ length: 0
                      Data_ free: 0
                 Auto_ increment: NULL
                    Create_ time: 2020-07-15 18:17:10
                    Update_ time: NULL
                     Check_ time: NULL
                       Collation: utf8_ general_ ci
                        Checksum: NULL
                 Create_ options: avg_ row_ length=5
            Limit_ storage_ size: 0
                  storage_ size: 284
               table_ data_ size: 0
                         Comment:
 local_ hash_ index_ file_ size: 0
global_ hash_ index_ file_ size: 0
               tablespace_ name: NULL
               tablespace_ path: NULL
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.10 SHOW FUNCTION STATUS

Function Description

Display the status of the function that has been successfully created.

Grammar format

```
SHOW FUNCTION STATUS [WHERE conditions];
```

Table 5123 Parameter Description

Parameter Name	explain
conditions	Filter conditions.

Example

Example 1: Display the status of a function that has been successfully created.

```
gbase> SHOW FUNCTION STATUS\G
*************************** 1. row ***************************
                    Vc: vc1
                    Db: demo
                  Name: hello
                  Type: FUNCTION
               Definer: root@%
              Modified: 2020-07-15 19:26:08
               Created: 2020-07-15 19:26:08
      Security_ type: DEFINER
               Comment:
character_ set_ client: utf8
collation_ connection: utf8_ general_ ci
  Database Collation: utf8_ general_ ci
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: Display the status of a successfully created function on vc1.

```
gbase> SHOW FUNCTION STATUS WHERE vc='vc1'\G
*************************** 1. row ***************************
                    Vc: vc1
                    Db: demo
                  Name: hello
                  Type: FUNCTION
               Definer: root@%
              Modified: 2020-07-15 19:26:08
               Created: 2020-07-15 19:26:08
      Security_ type: DEFINER
               Comment:
```

```
character_ set_ client: utf8
collation_ connection: utf8_ general_ ci
   Database Collation: utf8_ general_ ci
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.11 SHOW PROCEDURE STATUS

Function Description

Display the status of the successfully created stored procedure.

Grammar format

```
SHOW PROCEDURE STATUS [WHERE conditions];
```

Table 5124 Parameter Description

Parameter Name	explain
where conditions	Filter conditions.

Example

Example 1: Display the status of a successfully created stored procedure.

```
gbase>  SHOW PROCEDURE STATUS\G
*************************** 1. row ***************************
                    Vc: vc1
                    Db: demo
                  Name: proc_ one
                  Type: PROCEDURE
               Definer: root@%
              Modified: 2020-07-15 19:25:14
               Created: 2020-07-15 19:25:14
      Security_ type: DEFINER
               Comment:
character_ set_ client: utf8
collation_ connection: utf8_ general_ ci
   Database Collation: utf8_ general_ ci
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: Display the status of a successfully created stored procedure in the demo library of vc1.

```
gbase>  SHOW PROCEDURE STATUS where vc='vc1' and db='demo'\G
*************************** 1. row ***************************
                    Vc: vc1
                    Db: demo
                  Name: proc_ one
                  Type: PROCEDURE
               Definer: root@%
              Modified: 2020-07-15 19:25:14
               Created: 2020-07-15 19:25:14
      Security_ type: DEFINER
               Comment:
character_ set_ client: utf8
collation_ connection: utf8_ general_ ci
   Database Collation: utf8_ general_ ci
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.12 SHOW CREATE DATABASE

Function Description

Display the creation statement for the given database.

Grammar format

```
SHOW CREATE {DATABASE | SCHEMA} [vcname.]database_ name;
```

Table 5125 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name.

Example

Example 1: Display the statement to create a demo database.

```
gbase> SHOW CREATE DATABASE vc1.demo;
+----------+-----------------------------------------------------+
| Database | Create Database                                     |
+----------+-----------------------------------------------------+
| demo     | CREATE DATABASE "demo" DEFAULT CHARACTER SET utf8
|
+----------+-----------------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.13 SHOW CREATE TABLE

Function Description

Display the creation statement for the given table. This statement also applies to views. By parameter gbase_ show_ ident_ case_ Sensitive can control the casing of exported column names, which is consistent with the casing of column names in the source table structure by default. Please refer to the configuration parameters of Gnode in Chapter 7.6.3 for details.

Grammar format

```
SHOW [FULL] CREATE TABLE [vc_name.][database_name.]table_ name [for sync];
```

Table 5126 Parameter Description

Parameter Name	explain
FULL	Display more detailed table building statements, including TID, UID, COLUMN_ IDS and other information.
vc_ name	VC name, optional.
database_ name	Database name, optional.
for sync	When used in conjunction with the FULL keyword, TID and UID information will not be displayed

Example

Example 1: Display the statement to create a t table.

```
gbase>    SHOW CREATE TABLE t \G
*************************** 1. row ***************************
            Table: t
Create Table: CREATE TABLE "t" (
   "a" int(11) DEFAULT NULL
) ENGINE=EXPRESS DEFAULT CHARSET=utf8 TABLESPACE='sys_
tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: Create a statement that displays a multi column HASH distribution table

```
gbase> SHOW CREATE TABLE x1 \G
*************************** 1. row ***************************
            Table: x1
Create Table: CREATE TABLE "x1" (
   "entry_id" int(11) DEFAULT NULL,
   "id2" int(11) DEFAULT NULL,
   "id3" int(11) DEFAULT NULL,
   "id4" int(11) DEFAULT NULL
) ENGINE=EXPRESS DISTRIBUTED BY('id3','id4') DEFAULT
CHARSET=utf8 TABLESPACE='sys_ tablespace'
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.14 SHOW CREATE VIEW

Function Description

Display the creation statement for the given view.

Grammar format

```
SHOW CREATE VIEW [vc_name.][database_name.]view_ namel;
```

Table 5127 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
view_ name	View name.

Example

Example 1: Display creation t1_ The statement of the v view.

```
gbase> SHOW CREATE VIEW vc1.demo.t1_ v\G
*************************** 1. row ***************************
                       View: t1_ v
                Create View: CREATE ALGORITHM=TEMPTABLE
DEFINER="root"@"%" SQL SECURITY DEFINER VIEW "t1_v" AS select "t1".
"a" AS "a" from "t1"
character_ set_ client: utf8
collation_ connection: utf8_ general_ ci
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.15 SHOW CREATE FUNCTION

Function Description

Display the creation statement for the given custom function.

Grammar format

```
SHOW CREATE FUNCTION [vc_name.][database_name.]func_ name;
```

Table 5128 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
func_ name	Custom function name.

Example

Example 1: Display the statement to create a hello function.

```
gbase> show create function vc1.demo.hello\G
*************************** 1. row ***************************
                  Function: hello
                  sql_ mode: PIPES_ AS_ CONCAT,ANSI_ QUOTES,IGNORE_
SPACE,ONLY_ FULL_ GROUP_ BY,NO_ AUTO_ VALUE_ ON_
ZERO,STRICT_ ALL_ TABLES,NO_ ZERO_ IN_ DATE,NO_ ZERO_
DATE,NO_ AUTO_ CREATE_ USER,NO_ ENGINE_ SUBSTITUTION,PAD_
CHAR_ TO_ FULL_ LENGTH
        Create Function: CREATE DEFINER="root"@"%" FUNCTION "hello"(s
CHAR(20)) RETURNS char(50) CHARSET utf8
RETURN CONCAT('Hello, ',s,'!')
character_ set_ client: utf8
collation_ connection: utf8_ general_ ci
   Database Collation: utf8_ general_ ci
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.16 SHOW CREATE PROCEDURE

Function Description

Display the creation statement for the given stored procedure.

Grammar format

```
SHOW CREATE PROCEDURE [vc_name.][database_name.]proc_ name;
```

Table -5129 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
proc_ name	Stored procedure name.

Example

Example 1: Displaying the Create Stored Procedure proc_ 1 statement.

```
gbase> SHOW CREATE PROCEDURE vc1.demo.proc_ 1\G
*************************** 1. row ***************************
                  Procedure: proc_ one
                    sql_ mode: PIPES_ AS_ CONCAT,ANSI_ QUOTES,IGNORE_
SPACE,ONLY_ FULL_ GROUP_ BY,NO_ AUTO_ VALUE_ ON_
ZERO,STRICT_ ALL_ TABLES,NO_ ZERO_ IN_ DATE,NO_ ZERO_
DATE,NO_ AUTO_ CREATE_ USER,NO_ ENGINE_ SUBSTITUTION,PAD_
CHAR_ TO_ FULL_ LENGTH
       Create Procedure: CREATE DEFINER="root"@"%" PROCEDURE
"proc_1"()
begin
select 1;
end
character_ set_ client: utf8
collation_ connection: utf8_ general_ ci
   Database Collation: utf8_ general_ ci
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.17 SHOW CREATE SYNONYM

Function Description

Display the creation statement for the given private synonym.

Grammar format

```
SHOW CREATE SYNONYM [vc_name.][database_name.]syn_ name;
```

Table 5130 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
syn_ name	Synonymous name.

Example

Example 1: To view the creation statement of private synonym s1:

```
gbase> create synonym s1 for demo.t1;
Query OK, 0 rows affected (Elapsed: 00:00:00.04)

gbase> show create synonym vc1.demo.s1;
+-------------+-----------------------------------------+
| Synonym          | Create synonym                                          |
+-------------+-----------------------------------------+
| vc1.demo.s1 | CREATE SYNONYM vc1.demo.s1 FOR vc1.demo.t1 |
+-------------+-----------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.18 SHOW CREATE PUBLIC SYNONYM

Function Description

Display the creation statement for the given public synonym.

Grammar format

```
SHOW CREATE PUBLIC SYNONYM [vc_name.]sym_ name;
```

Table 5131 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
syn_ name	Synonymous name.

Example

Example 1: To view the creation statement of the public synonym s2:

```
gbase> CREATE PUBLIC SYNONYM s2 FOR demo.t1;
Query OK, 0 rows affected (Elapsed: 00:00:00.03)

gbase> SHOW CREATE PUBLIC SYNONYM s2;
+---------+-----------------------------------------------+
| Synonym | Create synonym                                |
+---------+-----------------------------------------------+
| vc1.s2  | CREATE PUBLIC SYNONYM vc1.s2 FOR vc1.demo.t1 |
+---------+-----------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)

gbase> SHOW CREATE PUBLIC SYNONYM vc1.s2;
+---------+-----------------------------------------------+
| Synonym | Create synonym                                |
+---------+-----------------------------------------------+
| vc1.s2  | CREATE PUBLIC SYNONYM vc1.s2 FOR vc1.demo.t1 |
+---------+-----------------------------------------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.19 SHOW GRANTS

Function Description

List the GRANT statements that allow permissions to be granted to a GBase 8a MPP Cluster user account.

Grammar format

```
SHOW GRANTS FOR {[user_name] | CURRENT_USER[()]};
```

Table 5132 Parameter Description

Parameter Name	explain
user_ name	user name. If not specified, there is no current user permission
CURRENT_ USER[()]	Current user.

Example

Example 1: Display the permissions of the root user (a statement granting permissions to the root user).

```
gbase> SHOW GRANTS FOR 'root'@'%'\G
*************************** 1. row ***************************
Grants for root@%: GRANT ALL PRIVILEGES ON *.*.* TO 'root'@'%' WITH
```

```
GRANT OPTION TASK_ PRIORITY 2
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.20 SHOW TABLE LOCKS

Function Description

Display table lock information.

Grammar format

```
SHOW TABLE LOCKS;
```

5.1.11.7.21 SHOW DISTRIBUTION

Function Description

List table information for the specified database, including VC name, database name, table name, and whether it is a replicated table.

Grammar format

```
SHOW DISTRIBUTION TABLES [FROM [vc_name.]database_ name] [LIKE 'pattern']
```

Table -5133 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name.
pattern	A string that can contain SQL '%' and '_' wildcards.

Example

Example 1: Display the table information of the demo library in vc1.

```
gbase> SHOW DISTRIBUTION TABLES FROM vc1.demo;
+--------+--------+--------+-------------+
| vcName | dbName | tbName | isReplicate |
+--------+--------+--------+-------------+
| vc1    | demo   | d1     | NO          |
| vc1    | demo   | th     | NO          |
| vc1    | demo   | t1     | NO          |
```

```
| vc1       | demo      | tr          | YES                 |
+--------+--------+--------+-------------+
4 rows in set (Elapsed: 00:00:00.00)
```

Example 2: Display table information starting with d under the demo library in vc1.

```
gbase> SHOW DISTRIBUTION TABLES FROM vc1.demo LIKE 'd%';
+--------+--------+--------+-------------+
| vcName | dbName | tbName | isReplicate |
+--------+--------+--------+-------------+
| vc1       | demo      | d1          | NO                  |
+--------+--------+--------+-------------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.22 SHOW OPEN TABLES

Function Description

List the tables opened under the specified database.

Grammar format

```
SHOW OPEN TABLES FROM [vc_name.]database_ name { [WHERE conditions] | [LIKE 'pattern']}
```

Table -5134 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name.
pattern	A string that can contain SQL '%' and '_' wildcards.
conditions	Filter conditions.

Example

Example 1: Display the table opened under the demo library in vc1.

```
gbase> SHOW OPEN TABLES FROM vc1.demo;
+----------+-------+--------+-------------+-----+
| Database | Table | In_ use | Name_ locked | VC    |
+----------+-------+--------+-------------+-----+
| demo         | d1       |          0 |                  0 | vc1 |
| demo         | t1       |          0 |                  0 | vc1 |
```

```
+----------+-------+--------+-------------+-----+
2 rows in set (Elapsed: 00:00:00.00)
```

Example 2: Display a table with a name starting with d opened under the demo library in vc1.

```
gbase> SHOW OPEN TABLES FROM demo LIKE 'd%';
+----------+-------+--------+-------------+-----+
| Database | Table | In_ use | Name_ locked | VC    |
+----------+-------+--------+-------------+-----+
| demo         | d1       |           0 |                   0 | vc1 |
+----------+-------+--------+-------------+-----+
1 row in set (Elapsed: 00:00:00.00)
```

Example 3: Displaying in under the demo library in vc1_ Open table with user 0.

```
gbase> SHOW OPEN TABLES FROM demo WHERE in_ use=0;
+----------+-------+--------+-------------+-----+
| Database | Table | In_ use | Name_ locked | VC    |
+----------+-------+--------+-------------+-----+
| demo         | d1       |           0 |                   0 | vc1 |
| demo         | t1       |           0 |                   0 | vc1 |
+----------+-------+--------+-------------+-----+
2 rows in set (Elapsed: 00:00:00.00)
```

5.1.11.7.23 SHOW PRIORITIES

Function Description

Display priority status, including cluster node name, priority number, priority on/off status, priority counterweight, priority control parameter description, and other information.

Grammar format

```
SHOW PRIORITIES [WHERE conditions]
```

Table -5135 Parameter Description

Parameter Name	explain
conditions	Filter conditions.

Example

Example 1: View the priority status of all nodes in the cluster.

```
gbase> show priorities;
+-----------+----------+--------+--------+-------------+
| Node_ name | priority | status | weight | description |
+-----------+----------+--------+--------+-------------+
| node1      |        0 | OFF    |      0 |             |
| node1      |        1 | OFF    |     20 |             |
| node1      |        2 | OFF    |     40 |             |
| node1      |        3 | OFF    |     80 |             |
| node2      |        0 | OFF    |      0 |             |
| node2      |        1 | OFF    |     20 |             |
| node2      |        2 | OFF    |     40 |             |
| node2      |        3 | OFF    |     80 |             |
| node3      |        0 | OFF    |      0 |             |
| node3      |        1 | OFF    |     20 |             |
| node3      |        2 | OFF    |     40 |             |
| node3      |        3 | OFF    |     80 |             |
| node4      |        0 | OFF    |      0 |             |
| node4      |        1 | OFF    |     20 |             |
| node4      |        2 | OFF    |     40 |             |
| node4      |        3 | OFF    |     80 |             |
+-----------+----------+--------+--------+-------------+
12 rows in set
```

Example 2: Viewing the priority status information of node1 node.

```
gbase> SHOW PRIORITIES WHERE "node_name" = 'node1';
+-----------+----------+--------+--------+-------------+
| Node_ name | priority | status | weight | description |
+-----------+----------+--------+--------+-------------+
| node1      |        0 | ON     |      0 |             |
| node1      |        1 | ON     |     20 |             |
| node1      |        2 | ON     |     40 |             |
| node1      |        3 | ON     |     80 |             |
+-----------+----------+--------+--------+-------------+
4 rows in set
```

Example 3: Viewing priority information with a status of ON.

```
gbase> SHOW PRIORITIES WHERE status ='ON';
+-----------+----------+--------+--------+-------------+
| Node_ name | priority | status | weight | description |
+-----------+----------+--------+--------+-------------+
| node1      |        0 | ON     |      0 |             |
| node1      |        1 | ON     |     20 |             |
| node1      |        2 | ON     |     40 |             |
| node1      |        3 | ON     |     80 |             |
| node2      |        0 | ON     |      0 |             |
| node2      |        1 | ON     |     20 |             |
| node2      |        2 | ON     |     40 |             |
| node2      |        3 | ON     |     80 |             |
| node3      |        0 | ON     |      0 |             |
| node3      |        1 | ON     |     20 |             |
| node3      |        2 | ON     |     40 |             |
| node3      |        3 | ON     |     80 |             |
| node4      |        0 | ON     |      0 |             |
| node4      |        1 | ON     |     20 |             |
| node4      |        2 | ON     |     40 |             |
| node4      |        3 | ON     |     80 |             |
+-----------+----------+--------+--------+-------------+
12 rows in set
```

Example 4: Turn off the service cgconfig stop of node1 node cgroup configuration service

```
gbase> SHOW PRIORITIES WHERE "node_name" = 'node1';
+-----------+----------+--------+--------+-------------+
| Node_ name | priority | status | weight | description |
+-----------+----------+--------+--------+-------------+
| node1      |        0 | OFF    |      0 |             |
| node1      |        1 | OFF    |     20 |             |
| node1      |        2 | OFF    |     40 |             |
| node1      |        3 | OFF    |     80 |             |
+-----------+----------+--------+--------+-------------+
4 rows in set
```

Example 5: Restart the cgroup configuration service of Node1 (service cgconfig start)

```
gbase> SHOW PRIORITIES WHERE "node_name" = 'node1';
+-----------+----------+--------+--------+-------------+
| Node_ name | priority | status | weight | description |
+-----------+----------+--------+--------+-------------+
| node1      |        0 | ON     |      0 |             |
| node1      |        1 | ON     |     20 |             |
| node1      |        2 | ON     |     40 |             |
| node1      |        3 | ON     |     80 |             |
+-----------+----------+--------+--------+-------------+
4 rows in set
```

5.1.11.7.24 SHOW PROCESSLIST

Function Description

SHOW PROCESS LIST displays the running threads. If you have SUPER permission, you can see all threads. Otherwise, users can only see their own threads (which are related to the GBase 8a MPP Cluster account they are using). GBase 8a MPP Cluster reserves an additional connection for accounts with SUPER privileges to ensure that administrators can always connect and verify the system (assuming that SUPER privileges are not granted to all users). This statement is very useful if the user receives an error message of "too many connections" and wants to understand what is happening.

Grammar format

```
SHOW [FULL] PROCESSLIST
```

Table 5136 Parameter Description

Parameter Name	explain
FULL	If you do not use the FULL keyword and only display the first 100 characters of each executing SQL statement, using the FULL keyword can display the content of the entire SQL statement, as well as the thread number used for the connection in the second column.

Table 5137 Display Information Column Description

Parameter Name	explain
Id	The ID number of the company level.
Tid	The thread number used for this connection.
User	Connected users.

Parameter Name	explain
Host	The host name of a TCP/IP connection makes it easier to see the running status of each client, in the form of host_ name:client_ port。
vc	Connected VC.
db	Database name.
Command	Connect to the SQL executed.
Time	The time of connection.
State	Connection status.
Info	Connection information.

Table -5138 State Column Status Description

Parameter Name	explain
Initialized	Indicates that the initialization of the event scheduler is complete
Waiting for cluster mutex	Indicates that the event scheduler is waiting for a cluster global lock, and only global mode schedulers have this state
Waiting for empty queue	There are no events to be executed in the event queue of the time scheduler, and the administrator is waiting to create a new event
Waiting for next activation	Indicates that the scheduler is waiting for events in the queue to arrive at the execution time.

Example

Example: Display running threads.

```
gbase> SHOW PROCESSLIST\G
*************************** 1. row ***************************
     Id: 1
   User: event_ scheduler
   Host: localhost
     vc: NULL
     db: NULL
Command: Daemon
   Time: 6698
  State: Waiting for next activation
   Info: NULL
1 rows in set (Elapsed: 00:00:00.00)
```

5.1.11.7.25 SHOW STATUS

Function Description

Provide status information.

Grammar format

```
SHOW [GLOBAL | SESSION] STATUS { [LIKE 'pattern'] | [WHERE conditions] }
```

Table 5139 Parameter Description

Parameter Name	explain
GLOBAL	Use the GLOBAL option to obtain the status values of all connections to GBase 8a MPP Cluster
SESSION	The SESSION option can obtain the status value of the current connection. If no options are used, the default value is SESSION. LOCAL and SESSION have the same meaning.
pattern	A string that can contain SQL '%' and '_' wildcards.
conditions	Filter conditions.

be careful

Some state variables only have global values, so whether using GLOBAL or SESSION, they can only obtain the same value.

Example

Example 1: Viewing status information.

```
gbase> SHOW STATUS;
+--------------------------------------+------------+
| Variable_ name                       | Value      |
+--------------------------------------+------------+
| Aborted_ clients                     | 0          |
| Aborted_ connects                    | 6          |
| Audit_ queries                       | 1          |
| Binlog_ cache_ disk_ use             | 0          |
| Binlog_ cache_ use                   | 0          |
| Bytes_ received                      | 2671       |
...
| cluster_ mode                        | NORMAL     |
| cluster_ state                       | ACTIVED    |
| local_ nodeid                        | 1930078400 |
+--------------------------------------+------------+
272 rows in set
```

Example 2: A statement with a LIKE clause only displays variables of matching types.

```
gbase> SHOW STATUS LIKE 'Key%';
+------------------------+-------+
| Variable_ name         | Value |
+------------------------+-------+
| Key_ blocks_ not_ flushed | 0  |
| Key_ blocks_ unused    | 6675  |
| Key_ blocks_ used      | 19    |
| Key_ read_ requests    | 503   |
| Key_ reads             | 4     |
| Key_ write_ requests   | 459   |
| Key_ writes            | 267   |
+------------------------+-------+
7 rows in set
```

5.1.11.7.26 SHOW VARIABLES

Function Description

Display the values of some GBase 8a MPP Cluster system variables.

Grammar format

```
SHOW [GLOBAL | SESSION] VARIABLES{ [LIKE 'pattern'] | [WHERE conditions] }
```

Table 5140 Parameter Description

Parameter Name	explain
GLOBAL	Can obtain variable values for new connections to GBase 8a MPP Cluster.
SESSION	Can obtain the current connected variable value. If the option is not used, the default value is SESSION. LOCAL and SESSION have the same meaning.
pattern	A string that can contain SQL '%' and '_' wildcards.
conditions	Filter conditions.

Example

Example 1: Using the LIKE clause, this statement only displays variables of matching types.

```
gbase> SHOW VARIABLES LIKE '%buffer%';
+-------------------------+-----------+
| Variable_ name                  | Value         |
+-------------------------+-----------+
| bulk_ insert_ buffer_ size | 8388608     |
| gbase_ buffer_ distgrby     | 16777216    |
| gbase_ buffer_ hgrby          | 16777216    |
| gbase_ buffer_ hj               | 16777216    |
| gbase_ buffer_ insert         | 268435456 |
| gbase_ buffer_ result         | 209715200 |
| gbase_ buffer_ rowset         | 8388608     |
| gbase_ buffer_ sj               | 16777216    |
| gbase_ buffer_ sort             | 16777216    |
| gssys_ sort_ buffer_ size    | 8388608     |
| join_ buffer_ size              | 131072       |
| key_ buffer_ size               | 8384512     |
| net_ buffer_ length             | 16384         |
| preload_ buffer_ size         | 32768         |
| read_ buffer_ size              | 131072       |
| read_ rnd_ buffer_ size       | 262144       |
| sort_ buffer_ size              | 2097144     |
| sql_ buffer_ result             | OFF             |
+-------------------------+-----------+
18 rows in set
```

5.1.11.7.27 SHOW ERRORS

Function Description

Display error messages generated by the last statement. When the last statement using a table does not generate a message, no message is displayed. Every time a new sentence is executed, the message list will be reset.

Grammar format

```
SHOW ERRORS [LIMIT [offset,] row_ count]
```

5.1.11.7.28 SHOW COUNT(*) ERRORS

Function Description

Display the number of error messages generated by the last statement, which can also be obtained from the error_ Obtaining the number of errors in the count variable.

Grammar format

```
SHOW COUNT(*) ERRORS;
SELECT @@error_ count;
```

5.1.11.7.29 SHOW WARNINGS

Function Description

Display warning and caution messages generated by the last statement. When the last statement using a table does not generate a message, no message is displayed. Every time a new sentence is executed, the message list will be reset.

Grammar format

```
SHOW WARNINGS [LIMIT [offset,] row_ count]
```

Example

Example 1: Viewing warning messages.

```
gbase> SHOW WARNINGS;
+---------+---------+-------------------------------------------------+
|       Level                 |       Code                  |       Message
|
+---------+---------+-------------------------------------------------+
| Warning    | 1051       | (GBA-02DD-0010) Unknown table 'test.no_ such_ table'|
+---------+---------+-------------------------------------------------+
1 row in set
```

5.1.11.7.30 SHOW COUNT (*) WARNINGS

Function Description

Display the number of warning and attention messages generated by the last statement.

Grammar format

```
SHOW COUNT(*) WARNINGS;
SELECT @@warning_ count;
```

be careful

- max_ error_ The count system variable controls the maximum number of errors, warnings, and attention messages that can be stored, with a default value of 64. Users can change the value of this variable to change the amount of information that can be stored. If max_ error_ The count system variable is set too small to store all information
- Add max_ error_ If count is set to 0, no warning information is stored. In this case, warning_ Count still indicates the number of warnings that have occurred, but does not store any warning content.

Example

Example 1: The alter table statement generates three warning messages, but due to max_ error_ The count value is 1, so only one warning is stored.

```
gbase> SHOW VARIABLES LIKE 'max_ error_ count';
+-----------------+-------+
| Variable_ name      | Value |
+-----------------+-------+
| max_ error_ count | 64        |
+-----------------+-------+
gbase> SET max_ error_ count=1;
Query OK, 0 rows affected
gbase> SELECT 1 > '6x' FROM dual;
+----------+
| 1 > '6x' |
+----------+
|               0 |
+----------+
1 row in set, 2 warnings
gbase> SELECT @@warning_ count;
+-----------------+
| @@warning_ count |
+-----------------+
|                               1 |
+-----------------+

gbase> SHOW WARNINGS;
+-------+------+
| Level | Code |
+-------+------+
| Note    | 1292 |
+-------+------+
+-----------------------------------------------------------+
| Message                                                                                         |
+-----------------------------------------------------------+
| 192.168.10.115:5050 - Truncated incorrect DOUBLE value: '6x' |
+-----------------------------------------------------------+
1 row in set
```

5.1.11.7.31 SHOW GCLUSTER ENTRY

Function Description

Used to obtain the node with the least number of connections in the cluster, in order to achieve load balancing at the connection layer.

Grammar Format Column Technology

```
SHOW GCLUSTER ENTRY;
```

Table 5141 Parameter Description

Parameter Name	explain
IP	The cluster has the minimum number of connected nodes' IPs.
Port	The GCluster service port number of the node with the minimum number of connections in the cluster.

Example

Example 1: View the information of the node with the least number of connections in the cluster.

```
gbase> SHOW GCLUSTER ENTRY;
+--------------+------+
| IP           | Port |
+--------------+------+
| 172.168.83.12| 5258 |
+--------------+------+
1 row in set (Elapsed: 00:00:00.00)
```

5.1.11.7.32 SHOW [GCLUSTER] NODES

Function Description

Used to obtain node information for coordinator or data clusters.

Grammar format

```
SHOW [GCLUSTER] NODES;
```

Example

Example 1: View the node information of the coordinator cluster.

```
gbase> SHOW GCLUSTER NODES;
+-----------+---------------+--------------+--------+-----------+
| Id        | ip            | name         | status | datastate |
+-----------+---------------+--------------+--------+-----------+
| 190032044 | 172.168.83.11 | coordinator1 | online |         0 |
| 206809260 | 172.168.83.12 | coordinator2 | online |         0 |
| 223586476 | 172.168.83.13 | coordinator3 | online |         0 |
+-----------+---------------+--------------+--------+-----------+
3 rows in set (Elapsed: 00:00:00.00)
```

Example 2: Viewing node information of a data cluster.

```
gbase> SHOW NODES\G
*************************** 1. row ***************************
          Id: 190032044
          ip: 172.168.83.11
        name: node1
  primary part: n1
duplicate part: n2
      status: online
   datastate: 0
*************************** 2. row ***************************
          Id: 206809260
          ip: 172.168.83.12
        name: node2
  primary part: n2
duplicate part: n1
      status: online
   datastate: 0
1
2 rows in set (Elapsed: 00:00:00.00)
```

5.1.11.8 EXPLAIN Display Query Plan

The EXPLAIN command of GBase 8a MPP Cluster displays the correct query plan. For CBO (Cost Based Optimization) plans, the evaluation results for each step are displayed, including cost, number of records, record width, and selection rate. Users can view the plan before executing SQL.

Grammar:

- To view the query plan for SELECT:

```
EXPLAIN/DESC [extended/partitions] SELECT……
```

- View CTE's query plan:

```
EXPLAIN/DESC [extended/partitions] WITH……SELECT
```

be careful

EXPLAIN is equivalent to DESC, so it can be interchanged to view the query plan of a SELECT.

5.1.11.8.1 Default output

When there are no Extended and Partitions after EXPLAIN, the default output is displayed.

The default output only outputs a simplified version of the query plan, mainly including the data redistribution method and the main operations of each step.

The detailed explanation is as follows:

```
gbase> explain select * from x2 where id2 > (select count(*) from x3);
+----+-----------+-----------+----------+----------------+----+-------+------+
| ID | MOTION    | OPERATION | TABLE    | CONDITION      | ROW | WIDTH | COST(TOTAL)
|
+----+-----------+-----------+----------+----------------+----+------+------------+
| 02 | [RESULT]     | SCAN          | x2[id4]     | id2{S} > &x1x&) | 16    | 16       | 1.05(3.26) |
| 01 | [SCALAR_1] | Step          | <00>        |                         | 0     | 0        | 0.52(2.21) |
|    |                  | AGG         |                |                         |       |         |              |
| 00 | [GATHER]     | Table         | x3[id4]     |                         | 0     | 0        | 1.69(1.69) |
|    |                  | AGG         |                |                         |      |          |             |
+----+-----------+-----------+----------+---------------+----+-------+------------+
5 row in set
```

Table -5142 provides a detailed explanation as follows:

column	explain
ID	The steps of the plan are executed from 00 and sequentially from bottom to top.
MOTION	The processing method for the execution results of this step: RESULT: Send the results to the client; GATHER: Send the results to the summary node; REDIST (...): The result HASH is redistributed, with the columns in parentheses for calculating HASH. If they are too long, they are truncated to two points; NO REDIST: The results are directly saved to the corresponding data shards without redistribution; BROADCAST: Result pull replication table; RAND REDIST: The results are randomly distributed to all nodes; SCALAR_ N: The result is scalar, and N is the number of the scalar subquery. If there is a reference in the condition, use the&xNx&method to reference it. For example, in step 01, the quantum query [SCALAR_1] is selected, and in step 02, the reference to the scalar in step 01 is: id2 {S}>&x1x&
OPERATIO N	SCAN: Single table scan and use conditional filtering of data; Table: Single table without filtering conditions; SubQueryN: Sub query, where N is automatically numbered; Step: Use the result of the previous step; INNER/DEFT/FULL JOIN: Connection operation; WHERE: The WHERE condition of the subquery; GROUP: Group operation; ORDER: Sort operation; Limit: Calculate Limit, OFFSET; AGG: distinct, aggregation operation; UNION/UNION ALL/MINUS/INTERSECT: UNION operation; For example, the meaning of 00 in the figure is: The left side of the first INNER JOIN is the second INNER JOIN; The left side of the second INNER JOIN is date_ Dim, this table is a replicated table with conditional filtering; The right side of the second INNER JOIN is salesreturns, which is a subquery Salesreturns. The subquery contains a UNION ALL: a catalog_ Sa.. And catalog_ UNION ALL of re; The right side of the first INNER JOIN is the catalog_ Page, which is a replicated table; After the first INNER JOIN, there is a GROUP operation; For each step (where the ID column is not empty, it is a step), there is an indentation of 1 space in each layer of the Operation.
TABLE	The table involved in the Operation only displays aliases and attributes, and is truncated to two points if it is too long; HASH distribution table: the HASH column is displayed in parentheses; Copy Table: Display [REP]; Random distribution table: display [DIS];

	Sub query: The Operation column displays SubQueryN, where N is a number used to distinguish different sub queries; 某个步骤的结果：OPERATION 列显示为 Step，本列显示为<N>其中 N 为用到的步骤的第一列 ID，表示该步骤的结果； For example: The step with ID 03 in the figure is a summary step, and the source table is a sub query of x; The x sub query is a UNION that involves SSR, CSR, and WSR; SSR, CSR, and WSR are all sub queries, and the results from 02, 00, and 01 are grouped by.
CONDITION	Display the conditions of the operation, such as the single table conditions of FILTER, the join conditions of JOIN, the contents of GROUP BY, ORDER BY, and LIMITED. If a certain condition may use an index, it will be annotated after the indexed column, such as: ● Where t1. a {S}>10 indicates the possibility of using t1. a's smart index; ● WHERE t1. a {H}=10 indicates that the HASH index of t1. a may be used; ● WHERE t1. a {H}=t2. b indicates that the HASH index of t1. a may be used; ● WHERE t1. a {H}=t2. b {H} indicates that the HASH index of t1. a and t2. b may be used; ● Where contains (t1. a {F}...) indicates that a full text index of t1. a may be used. Only a single column of a physical table contains an index: ● Physical form columns may use smart indexes for>,<,>=,<=,=; ● The physical form column may use HASH indexes, including constant and JOIN equivalents; ● The contains function of a physical table may use full-text indexing.
ROW，WIDTH，COST (TOTAL)	Display the content of cost evaluation, including the number of rows ROW, row width WIDTH, cost of this step, and overall cost COST. The cost does not include the cost of data movement for the results. The values in the figure are not real data and are for reference only. Note: When some tables or columns lack statistical information, cost related content will not be displayed. Instead, those tables or columns that lack statistical information, such as x2 and x3 tables, will not be displayed with ROW, WIDTH, and COST columns. Instead, the NO STA Tab/Col column will be displayed, listing the missing statistical information related tables x2 and x3, as shown below: **gbase> explain select * from x2 where id2 > (select count(*) from x3);** +----+-----------+-----------+----------+---------------+----+-------+------+ | ID | MOTION | OPERATION | TABLE | CONDITION | **NO STA Tab/Col**|

```
+----+-----------+-----------+----------+---------------+----+-------+
| 02 | [RESULT]     | SCAN            | x2[id4] | (id2{S} > &x1x&) |                     |
| 01 | [SCALAR_1] | Step                  | <00>    |                         |                     |
|      |                                  | AGG                         |               |                            |
|
| 00 | [GATHER]     | Table                   | x3[id4]     |                         | x3               |
|     |                       | AGG                    |               |                         | x2               |
+----+-----------+-----------+----------+---------------+----+-------+5 row in set
```

5.1.11.8.2 Extended output

When there is an Extended after EXPLAIN, the extended output format of the query plan is displayed.

- Example 1:

```
EXPLAIN EXTENDED SELECT x1.id2 FROM x1, x2 WHERE x1.id2 = x2.id3
AND x1.id3 IN (SELECT id2 FROM x2 WHERE x2.id4 > 10);
```

Query Plan:

```
QueryPlan:

 Uncorrelated SUB plan

==================================================

==================================================
QueryPlan:

++++++++++++++++++++++++++++++++++++++++++++++++++
Leaf type: REGULAR STEP
Need combine: false
    Target temp table: _ tmp_ 2030479552_ 19_ t160_ 1_ 1496193667_ S
【 Target temporary table name 】
Temp table definition: CREATE TABLE `gctmpdb`._ tmp_ 2030479552_ 19_
t160_ 1_ 1496193667_ s AS SELECT /*192.168.6.121_ 19_ 15_ 2017-05-31_
09:24:31*/ /*+ TID('131280') */ DISTINCT `regress_ db_ link.x2`.`id2` AS `id2`
FROM `regress_ db_ link`.` x2` `regress_ db_ Link. x2 ` WHERE
('regress_db_link. x2 `. id4 `>10) LIMITED 0 [Create SQL for target table]
```

Optimization:

Query String: SELECT /*192.168.6.121_ 19_ 15_ 2017-05-31_ 09:24:31*/ /*+ TID('131280') */ DISTINCT `regress_ db_ link.x2`.`id2` AS `id2` FROM `regress_ db_ link`.` x2` `regress_ db_ Link. x2 ` WHERE ('regress_db_link. x2 `. id4 `>10) 【 Query statement 】

[Index Content] May use index: ` register_ db_ link`.` x2`.` id4`{Smart Index}

Cost Content

CostInfo: Start(0), Run(0.11), Selectivity(0.578947), Width(8), Rows(11)

==========

end SUB plan

++

Leaf type: REGULAR STEP

Need combiner: false

Target temp table: _ tmp_ 2030479552_ 19_ t160_ 2_ 1496193667_ s

Temp table definition: CREATE TABLE `gctmpdb`._ tmp_ 2030479552_ 19_ t160_ 2_ 1496193667_ s AS SELECT /*192.168.6.121_ 19_ 15_ 2017-05-31_ 09:24:31*/ /*+ TID('131280') */ `regress_ db_ link.x1`.`id2` AS `id2` FROM `regress_ db_ link`.` x1` `regress_ db_ link.x1` WHERE `regress_ db_ link.x1`.`id3` IN (SELECT `id2` FROM `gctmpdb`.`_tmp_2030479552_19_t160_1_1496193667_s`) LIMIT 0

Optimization:

Query String: SELECT /*192.168.6.121_ 19_ 15_ 2017-05-31_ 09:24:31*/ /*+ TID('131280') */ `regress_ db_ link.x1`.`id2` AS `id2` FROM `regress_ db_ link`.` x1` `regress_ db_ link.x1` WHERE `regress_ db_ link.x1`.`id3` IN (SELECT `id2` FROM `gctmpdb`.`_tmp_2030479552_19_t160_1_1496193667_s`)

Cost Content

CostInfo: Start(0), Run(0.07), Selectivity(0.291667), Width(8), Rows(7)

++

Leaf type: REGULAR STEP

Need combiner: false

Target temp table: _ tmp_ 2030479552_ 19_ t160_ 3_ 1496193667_ s

Temp table definition: CREATE TABLE `gctmpdb`._ tmp_ 2030479552_ 19_ t160_ 3_ 1496193667_ s AS SELECT /*192.168.6.121_ 19_ 15_ 2017-05-31_ 09:24:31*/ /*+ TID('131280') */ `_ tmp_ 2030479552_ 19_ t160_ 2_ 1496193667_ s`.` id2` AS `id2` FROM `gctmpdb`._ tmp_ 2030479552_ 19_ t160_ 2_

1496193667_ s INNER JOIN `regress_ db_ link`.` x2` `regress_ db_ link.x2` ON (`_tmp_2030479552_19_t160_2_1496193667_s`.`id2` = `regress_db_link.x2`.`id3`) LIMIT 0

Optimization:

Query String: SELECT /*192.168.6.121_ 19_ 15_ 2017-05-31_ 09:24:31*/ /*+ TID('131280') */ `_ tmp_ 2030479552_ 19_ t160_ 2_ 1496193667_ s`.` id2` AS `id2` FROM `gctmpdb`._ tmp_ 2030479552_ 19_ t160_ 2_ 1496193667_ s INNER JOIN `regress_ db_ link`.` x2` `regress_ db_ link.x2` ON (`_tmp_2030479552_19_t160_2_1496193667_s`.`id2` = `regress_db_link.x2`.`id3`)

Cost Content

CostInfo: Start(0.526), Run(0.52), Selectivity(0.0789474), Width(12), Rows(10)

==========

Uncorrelated Subplan:

CExecStep

IsQueryFinalStep=0 [not the last step of the query]

DestType=1 [Results sent to all nodes]

AStepDetail 【 8a All node attributes 】

IsProducer=1 [is a producer]

IsConsumer=1 [is a consumer]

ProducerDistID=1 [Producer distribution ID is 1]

ConsumerDistID=1 [Consumer distribution ID is 1]

IsSingleHashNode=0 [Non single node hash optimization]

IsHashRedist=0 [not hash redistribution]

isGroupHashRedist = 0

IsAllTableAreHashTmpDist=0 [Not all source tables are HASH temporary tables]

IsExistsHashReditTable=0 [The source table does not have a HASH temporary table]

queryString = SELECT /*192.168.6.121_ 19_ 15_ 2017-05-31_ 09:24:31*/ /*+ TID('131280') */ DISTINCT `regress_ db_ link.x2`.`id2` AS `id2` FROM `regress_ db_ link`.` x2` `regress_ db_ link.x2` WHERE

```
(`regress_db_link.x2`.`id4` > 10)

 targetTable = `gctmpdb`._ tmp_ 2030479552_ 19_ t160_ 1_ 1496193667_ s

 targetSchema = CREATE TABLE `gctmpdb`._ tmp_ 2030479552_ 19_ t160_ 1_ 1496193667_ s AS SELECT /*192.168.6.121_ 19_ 15_ 2017-05-31_ 09:24:31*/ /*+ TID('131280') */ DISTINCT `regress_ db_ link.x2`.`id2` AS `id2` FROM `regress_ db_ link`.` x2` `regress_ db_ link.x2` WHERE (`regress_db_link.x2`.`id4` > 10) LIMIT 0



 CExecStep

 --------------------------

 isQueryFinalStep = 0

 DestType = 1


 aStepDetail

 ---------------

 isProducer = 1

 isConsumer = 1

 producerDistID = 1

 consumerDistID = 1

 isSingleHashNode = 0

 isHashRedist = 0

 isGroupHashRedist = 0

 isAllTableAreHashTmpDist = 0

 isExistsHashReditTable = 0

 queryString = SELECT /*192.168.6.121_ 19_ 15_ 2017-05-31_ 09:24:31*/ /*+ TID('131280') */ `regress_ db_ link.x1`.`id2` AS `id2` FROM `regress_ db_ link`.` x1` `regress_ db_ link.x1` WHERE `regress_ db_ link.x1`.`id3` IN (SELECT `id2` FROM `gctmpdb`.`_tmp_2030479552_19_t160_1_1496193667_s`)

 targetTable = `gctmpdb`._ tmp_ 2030479552_ 19_ t160_ 2_ 1496193667_ s

 targetSchema = CREATE TABLE `gctmpdb`._ tmp_ 2030479552_ 19_ t160_ 2_ 1496193667_ s AS SELECT /*192.168.6.121_ 19_ 15_ 2017-05-31_ 09:24:31*/ /*+ TID('131280') */ `regress_ db_ link.x1`.`id2` AS `id2` FROM `regress_ db_ link`.` x1` `regress_ db_ link.x1` WHERE `regress_ db_ link.x1`.`id3` IN (SELECT `id2` FROM `gctmpdb`.`_tmp_2030479552_19_t160_1_1496193667_s`) LIMIT 0


 Step Drop List = `gctmpdb`._ tmp_ 2030479552_ 19_ t160_ 1_ 1496193667_ s
```

```
CExecStep
--------------------------
isQueryFinalStep = 0
DestType = 0

aStepDetail
---------------
isProducer = 1
isConsumer = 0
producerDistID = 1
consumerDistID = 1
isSingleHashNode = 0
isHashRedist = 0
isGroupHashRedist = 0
isAllTableAreHashTmpDist = 0
isExistsHashReditTable = 0
queryString = SELECT /*192.168.6.121_ 19_ 15_ 2017-05-31_ 09:24:31*/ /*+ TID('131280') */ `_ tmp_ 2030479552_ 19_ t160_ 2_ 1496193667_ s`.` id2` AS `id2` FROM `gctmpdb`._ tmp_ 2030479552_ 19_ t160_ 2_ 1496193667_ s INNER JOIN `regress_ db_ link`.` x2` `regress_ db_ link.x2` ON (`_tmp_2030479552_19_t160_2_1496193667_s`.`id2` = `regress_db_link.x2`.`id3`)

Step Drop List = `gctmpdb`._ tmp_ 2030479552_ 19_ t160_ 2_ 1496193667_ s
```

- Example 2:

```
explain extended select x1.id2 from x1, x2, x3 where x1.id2 = x2.id3 and x1.id3 = x3.id2;
```

Query Plan:

```
QueryPlan:

++++++++++++++++++++++++++++++++++++++++++++++++
Leaf type: REGULAR STEP
Need combiner: false
Target temp table: _ tmp_ 2030479552_ 19_ t161_ 1_ 1496193667_ s
Temp table definition: CREATE TABLE `gctmpdb`._ tmp_ 2030479552_ 19 _ t161_ 1_ 1496193667_ s AS SELECT /*192.168.6.121_ 19_ 16_ 2017-05-31_ 09:56:34*/ /*+ TID('131281') */ `regress_ db_ link.x3`.`id2` AS `id2` F
```

ROM `regress_ db_ link`.` x3` `regress_ db_ link.x3` LIMIT 0

Optimization:

Query String: SELECT /*192.168.6.121_ 19_ 16_ 2017-05-31_ 09:56:34*/ /*+ TID('131281') */ `regress_ db_ link.x3`.`id2` AS `id2` FROM `regress_ db_ link`.` x3` `regress_ db_ link.x3`

CostInfo: Start(0), Run(0.13), Selectivity(1), Width(4), Rows(13)

++

Leaf type: REGULAR STEP

Need combiner: false

Target temp table: _ tmp_ rht_ 2030479552_ 19_ t161_ 2_ 1496193667_ s

Temp table definition: CREATE TABLE `gctmpdb`._ tmp_ rht_ 2030479552_ 19_ t161_ 2_ 1496193667_ s AS SELECT /*192.168.6.121_ 19_ 16_ 2017-05-31_ 09:56:34*/ /*+ TID('131281') */ `regress_ db_ link.x1`.`id2` AS `id2` FROM `regress_ db_ link`.` x1` `regress_ db_ link.x1` INNER JOIN `gctmpdb`._ tmp_ 2030479552_ 19_ t161_ 1_ 1496193667_ s ON (`regress_db_link.x1`.`id3` = `_tmp_2030479552_19_t161_1_1496193667_s`.`id2`) LIMIT 0

Optimization: {hash redist} [is the HASH redistribution step]

Hash Redist Indexes: 1

Query String: SELECT /*192.168.6.121_ 19_ 16_ 2017-05-31_ 09:56:34*/ /*+ TID('131281') */ `regress_ db_ link.x1`.`id2` AS `id2` FROM `regress_ db_ link`.` x1` `regress_ db_ link.x1` INNER JOIN `gctmpdb`._ tmp_ 2030479552_ 19_ t161_ 1_ 1496193667_ s ON (`regress_db_link.x1`.`id3` = `_tmp_2030479552_19_t161_1_1496193667_s`.`id2`)

CostInfo: Start(0.812), Run(0.74), Selectivity(0.0608974), Width(12), Rows(19)

++

Leaf type: REGULAR STEP

Need combiner: false

Target temp table: _ tmp_ rht_ 2030479552_ 19_ t161_ 3_ 1496193667_ s

Temp table definition: CREATE TABLE `gctmpdb`._ tmp_ rht_ 2030479552_ 19_ t161_ 3_ 1496193667_ s AS SELECT /*192.168.6.121_ 19_ 16_ 2017-05-31_ 09:56:34*/ /*+ TID('131281') */ `regress_ db_ link.x2`.`id3` AS `id3` FROM `regress_ db_ link`.` x2` `regress_ db_ link.x2` LIMIT 0

```
            Optimization: {hash redist}

  Hash Redist Indexes: 1

            Query String: SELECT /*192.168.6.121_ 19_ 16_ 2017-05-31_ 09:
56:34*/ /*+ TID('131281') */ `regress_ db_ link.x2`.`id3` AS `id3` FROM `r
egress_ db_ link`.` x2` `regress_ db_ link.x2`


             CostInfo: Start(0), Run(0.19), Selectivity(1), Width(4), Rows(19)



++++++++++++++++++++++++++++++++++++++++++++++++

              Leaf type: REGULAR STEP

         Need combiner: false

    Target temp table: _ tmp_ 2030479552_ 19_ t161_ 4_ 1496193667_ s

Temp table definition: CREATE TABLE `gctmpdb`._ tmp_ 2030479552_ 19
_ t161_ 4_ 1496193667_ s AS SELECT /*192.168.6.121_ 19_ 16_ 2017-05-
31_ 09:56:34*/ /*+ TID('131281') */ `_ tmp_ rht_ 2030479552_ 19_ t161_ 2
_ 1496193667_ s`.` id2` AS `id2` FROM `gctmpdb`._ tmp_ rht_ 203047955
2_ 19_ t161_ 2_ 1496193667_ s INNER JOIN `gctmpdb`._ tmp_ rht_ 2030
479552_ 19_ t161_ 3_ 1496193667_ s ON (`_tmp_rht_2030479552_19_t161_
2_1496193667_s`.`id2` = `_tmp_rht_2030479552_19_t161_3_1496193667_s`.`id
3`) LIMIT 0

            Optimization:

            Query String: SELECT /*192.168.6.121_ 19_ 16_ 2017-05-31_ 09:
56:34*/ /*+ TID('131281') */ `_ tmp_ rht_ 2030479552_ 19_ t161_ 2_ 1496
193667_ s`.` id2` AS `id2` FROM `gctmpdb`._ tmp_ rht_ 2030479552_ 19_
 t161_ 2_ 1496193667_ s INNER JOIN `gctmpdb`._ tmp_ rht_ 2030479552
_ 19_ t161_ 3_ 1496193667_ s ON (`_tmp_rht_2030479552_19_t161_2_1496
193667_s`.`id2` = `_tmp_rht_2030479552_19_t161_3_1496193667_s`.`id3`)


             CostInfo: Start(1.40733), Run(1.31), Selectivity(0.0789474), Wi
dth(16), Rows(28)
```

5.1.11.8.2.1Query Plan Section

- There are two types of Leaf types for each step:
  - REGULAR STEP indicates execution at all nodes;
  - COMBINER STEP indicates execution at the summary node.
- Does this step require a need combiner? If so, it is true and there will be a COMBINER STEP using the target table for this step; If not required, it is false.
- The SQL statements executed in each step, including the statement to create the target temporary table (Temp table definition) and the query statement (Query String), for example:

```
Temp table definition: CREATE TABLE `gctmpdb`._ tmp_ rht_ 203047
9552_ 5_ t21_ 1_ 1494478737_ s AS SELECT /*192.168.6.121_ 5_ 53_ 20
17-05-11_ 16:17:05*/ /*+ TID('23') */ `lcg.x2`.`id2` AS `id2`, `lcg.x2`.`id3`
AS `id3`, `lcg.x2`.`dd` AS `dd` FROM `lcg`.` x2` `lcg.x2` LIMIT 0


Query String: SELECT /*192.168.6.121_ 5_ 53_ 2017-05-11_ 16:17:05*/
 /*+ TID('23') */ `lcg.x2`.`id2` AS `id2`, `lcg.x2`.`id3` AS `id3`, `lcg.x2`.`dd
` AS `dd` FROM `lcg`.` x2` `lcg.x2`
```

This may include comments and hints:

/*192.168.6.121_ 5_ 53_ 2017-05-11_ 16: 17:05 */is a comment

/*+TID ('23 ') */is the task ID

- The name of the target temporary table for each step, for example:

```
Target temp table: _ tmp_ rht_ 2030479552_ 5_ t21_ 1_ 1494478737_ s
```

  - _ tmp_ rht_: The beginning is the distribution table
  - _ Tmp: The table at the beginning is a replicated table
  - 2030479552: Originating node node id
  - 5: Thread ID (thread ->thread_id)
  - T21: Query ID
  - 1: Temporary Table Number
  - 1494478737: Timestamp
  - s: Temporary Table Suffix

- Is each step a HASH redistribution or a pull replication table.

  If it is a HASH redistribution, indicate the expression used to calculate the HASH value (Hash Redist Indexes), represented by an integer value N. When N is greater than 0, it represents the Nth expression of the Query String projection column (starting from 1).

  - When N equals 0, it represents a random distribution;
  - When N is equal to -10, it means that the query results directly fall on the shard without redistribution. This situation is usually used when reusing temporary tables. The results of the reused temporary tables are not redistributed, but directly fall on the nodes where the results are calculated.
  - For example:

    ```
    Optimization: {hash redist}
    Hash Redist Indexes: 1
    ```

```
Optimization: {rand redist}
Hash Redist Indexes: 0
```

```
Optimization: {no redist}
Hash Redist Indexes: -10
```

Without the above explanation, it is a pull copy table or summary table.

- Possible indexes to use

  List single columns that may use indexes, with index types including:

  - {Smart Index}: Smart index, used during scanning;
  - {Hash Index}: Hash index, used for equivalence comparison;
  - {Full Text}: Full text index, used when Contains.

  For example:

```
May used index: `regress_ db_ link.x1`.`entry_ id`{Smart Index}  `regress_ db_
link.x1`.`id2`{Hash Index}
```

- If it is a cost evaluation plan, output the start cost (Start), run cost (Run), selection rate (Selectivity), record width (Width), and number of results (Rows) for each step:

  - Startup cost: the cost of data redistribution and pulling replicated tables;
  - Running cost: Single table scanning, JOIN cost;
  - Selection rate: for a single table, represents the proportion of records filtered by the filtering criteria to the entire table;

    For JOIN, represents the proportion of records filtered by JOIN conditions to the Cartesian product.
  - Record width: represents the sum of the data lengths of all projected columns in this step. For fixed length types, use type length, variable length types, and use the average width of data in statistical information.

```
CostInfo: Start(0), Run(10), Selectivity(1), Width(16), Rows(1000)
```

5.1.11.8.2.2Execution Plan Section

The main content of the execution plan is consistent with the query plan, and some other attributes have been added, as shown in the table below:

The specific attributes of Table 5143 are as follows:

attribute	meaning
isQueryFinalStep	Whether to query the last step of the step

DestType	Target type: 1 represents all nodes, 2 represents summary nodes, and 3 represents a certain node (indicating that all SQL using the target table uses replicated tables). Other nodes are not used
isProducer	Producer or not: 0 No, 1 Yes
isConsumer	Consumer: 0 No, 1 Yes
producerDistID	Producer credit ID
consumerDistID	Consumer distribution ID
isSingleHashNode	Single node HASH optimization, 0 is not, 1 is
isHashRedist	HASH redistribution step, 0 is not, 1 is
Hash Redist Indexes	Expression subscript for HASH redistribution
isGroupHashRedist	not used
isAllTableAreHashTmpDist	Are all source tables HASH redistributed temporary tables. 0 is not, 1 is
isExistsHashReditTable	Is there a HASH redistribution temporary table in the source table. 0 is not, 1 is
queryString	SQL for this step
targetTable	Target Table
targetSchema	Target Table Creation Statement
[Step] DropList	List of temporary tables that can be deleted after this step is completed

5.1.11.8.3 Partitions output

When there are partitions after explain, a tree output is displayed.

The partitions method outputs a tree shaped query plan, which mainly includes the data redistribution method and the main operations of each step. The displayed content is similar to the default method.

- Example: TPC-DS SQL-5 plan:

```
+------------------------------------------------------------------------------+
|PlanTree
+------------------------------------------------------------------------------+
|     04:     [RESULT]     (row=2     width=26     cost=0.00693147(479021))
|
|                                                    Step     :     <03>
|
|                              GROUP   BY      ROLLUP   ((channel),   (id))
|
|                                  ORDER   BY   channel   ASC,   id   ASC
|
|                                                      LIMIT      100
```

```
|
|     --->03:     [GATHER]     (row=0     width=26     cost=250451(479021))
|
|                                                  Subquery    2    :    x
|
|
[                                                                |
|                                                Subquery    3    :    ssr
|
|                                                          Step    :    <02>
|
|                                                GROUP    BY    s_    store_    id
|
|                                                                          ]
|
|                                                          UNION     ALL
|
|
[                                                                |
|                                                Subquery    7    :    csr
|
|                                                          Step    :    <00>
|
|                                          GROUP    BY    cp_    catalog_    page_    id
|
|                                                                          ]
|
|                                                          UNION     ALL
|
|
[                                                                |
|                                                Subquery    11    :    wsr
|
|                                                          Step    :    <01>
|
|                                                GROUP    BY    web_    site_    id
|
|                                                                          ]
|
|                                              GROUP     BY     channel,     id
|
|                ---->02:    [REDIST]    (row=18    width=48    cost=128146(228570))
|
```

```
|                                                Redist Key : (s_store_id)
|
|                                                        INNER JOIN
|
|                                               ON (store_sk = s_store_sk)
|
|                                                        INNER JOIN
|
|                                             ON (date_sk = d_date_sk)
|
|                                                Table : tpcds.date_ dim
|
|                                                        Replicated
|
|                          WHERE (d_date BETWEEN cast('1998-08-04' as
date)AND
Adate_ add(cast('1998-08-04' as date), INTERV
|
|                                           WHERE AL 14 DAY))
|
|                                          Subquery 4 : salesreturns
|
|
[                                                  |
|                                                Table : tpcds.store_ sales
|
|                                                 Hash Key : ss_ item_ sk
|
|                                                                        ]
|
|                                                        UNION ALL
|
|
[                                                  |
|                                               Table : tpcds.store_ returns
|
|                                                 Hash Key : sr_ item_ sk
|
|                                                                        ]
|
|                                                    Table : tpcds.store
|
|                                                        Replicated
```

```
|
|                                                GROUP BY s_ store_ id
|
|              --->01: [REDIST] (row=45 width=48 cost=32897.2(100425))
|
|                                              Redist Key : (web_site_id)
|
|                                                          INNER JOIN
|
|                                       ON (wsr_web_site_sk = web_site_sk)
|
|                                                          INNER JOIN
|
|                                              ON (date_sk = d_date_sk)
|
|                                                 Table : tpcds.date_ dim
|
|                                                                Replicated
|
|                             WHERE (d_date BETWEEN cast('1998-08-04' as
date)
 AND date_ add(cast('1998-08-04' as date), INTERV
|
|                                              WHERE AL 14 DAY))
|
|                                              Subquery 12 : salesreturns
|
|
[                                                  |
|                                                 Table : tpcds.web_ sales
|
|                                                Hash Key : ws_ item_ sk
|
|                                                                        ]
|
|                                                          UNION ALL
|
|
[                                                  |
|                                                          LEFT JOIN
|
|                                       ON (wr_item_sk = ws_item_sk) AND
(wr_order
```

```
_ number = ws_ order_ number)
|
| Table : tpcds.web_ returns
|
| Hash Key : wr_ item_ sk
|
| Table : tpcds.web_ sales
|
| Hash Key : ws_ item_ sk
|
| ]
|
| Table : tpcds.web_ site
|
| Replicated
|
| GROUP BY web_ site_ id
|
| --->00: [REDIST] (row=35154 width=48 cost=67527.5(67527.5))
|
| Redist Key : (cp_catalog_page_id)
|
| INNER JOIN
|
| ON (page_sk = cp_catalog_page_sk)
|
| INNER JOIN
|
| ON (date_sk = d_date_sk)
|
| Table : tpcds.date_ dim
|
| Replicated
|
| WHERE (d_date BETWEEN cast('1998-08-04' as
date)
AND date_ add(cast('1998-08-04' as date), INTERV
|
| WHERE AL 14 DAY))
|
| Subquery 8 : salesreturns
|
|
```

```
[                                                                    |
|                                                 Table : tpcds.catalog_ sales
|
|                                                   Hash Key : cs_ item_ sk
|
|                                                                        ]
|
|                                                              UNION ALL
|
|
[                                                                    |
|                                                Table : tpcds.catalog_ returns
|
|                                                   Hash Key : cr_ item_ sk
|
|                                                                        ]
|
|                                                Table : tpcds.catalog_ page
|
|                                                                 Replicated
|
|                                       GROUP BY cp_ catalog_ page_ id
|
+-------------------------------------------------------------------------------+
98 rows in set (Elapsed: 00:00:00.29)
```

- When statistical information is missing, the cost evaluation content is not displayed, and the missing statistical information for which tables or columns is displayed in the header, as follows:

```
gbase> explain select * from x2 where id2 > (select count(*) from x3);
+-----------------------------------------+
|  PlanTree   [ NO STAT Tab/Col:x3 x2]          |
+-----------------------------------------+
| 02 : [RESULT]                                  |
|            Table : regress_ db_ link.x2          |
|                 Hash Key : id4                   |
|           WHERE (id2 > &x1x&)                      |
```

```
| --01: [SCALAR 1]                                |
|                  Step : <00>                    |
|                  AGG                            |
|       --00: [GATHER]                            |
|                        Table : regress_ db_ link.x3 |
|                             Hash Key : id4       |
|                        AGG                            |
+-----------------------------------------------+
11 row in set
```

5.2 Data integration and data management

5.2.1 Data loading

5.2.1.1 Data Server Configuration

The loading function of GBase 8a MPP Cluster V9.5.2. X supports pulling data from a universal data server, supporting multiple protocols such as FTP/HTTP/hdfs/sftp, and supporting Kafka clusters as data sources to load data.

The following is a brief description of the configuration methods for four common file servers on the Red Hat Enterprise Linux 6.2 platform: FTP, HTTP, HDFS, and SFTP.

5.2.1.1.1 FTP Server Configuration

Use vsftp to build an FTP server.

1) Check if vsftpd is installed

```
# rpm -qa vsftpd
vsftpd-2.2.2-6.el6_ 0.1.x86_ sixty-four
```

2) Install vsftpd

```
# rpm -ivh vsftpd-2.2.2-6.el6_ 0.1.x86_ 64.rpm
```

3) Modify the default configuration of the FTP server

 # vim /etc/vsftpd/vsftpd.conf

```
#Indicates that anonymous users are allowed to log in (default is YES)
anonymous_ enable=YES

#Indicates that local users are allowed to log in (default is YES)
local_ enable=YES

#Indicates open write access to local users (default YES, if used only as a loading file server, can be changed to NO)
write_ enable=NO

#Set the file generation mask for local users (the default file generation mask for local users is 077, which can be changed to 022)
local_ umask=022

#Allow anonymous FTP users to upload files (default is NO)
#anon_ upload_ enable=YES

#Allow anonymous FTP users to create directories (default is NO)
#anon_ mkdir_ write_ enable=YES

#Enable connection requests for FTP data ports (default is YES)
connect_ from_ port_ 20=YES

#The configuration file name using PAM authentication, located in the/etc/pam. d directory
pam_ service_ name=vsftpd

#Whether to use the userlist file to control access to the FTP server
userlist_ enable=YES

#Setting Prohibited Access Files or Directories
#deny_ file={*.mp3,*.mov,.private}
```

```
#Set hidden files or directories
#hide_ file={*.mp3,.hidden,hide*,h?}

#Set the FTP passive mode open port range (default to 0, indicating any available
port)
pasv_ min_ port=20001
pasv_ max_ port=21000

#Set the maximum allowed number of customer connections (default is 2000)
max_ clients=2000

#Set the maximum number of customer connections allowed on each IP (default is
50)
max_ per_ ip=50

#Connection timeout for passive transmission method (default is 60)
accept_ timeout=60

#Connection timeout for active transmission method (default is 60)
connect_ timeout=60

#Data transmission timeout in no progress state (default is 300)
data_ connection_ timeout=300

#Idle connection timeout (default is 300)
idle_ session_ timeout=300

#Whether to use system call sendfile to optimize transmission (default is YES, set
to NO when using network disks such as nfs)
use_ sendfile=YES

#Set the home directory for non anonymous login users
#local_ root=/var/ftp/pub

For more configurations, please refer to the vsftpd.com document
```

```
# man vsftpd.conf
```

When the maximum number of concurrent loading tasks in the cluster is N and the maximum number of single loading tasks (max_data_processors) is M, the minimum and recommended values for some parameters are as follows:

Table -5144 Parameter Values

Parameter Name	Default value	minimum value	Recommended value
max_ clients	two thousand	M*N	M*N*2
max_ per_ ip	fifty	N	N*2
pasv_ min_ port	0	max-min : M*N	max-min : M*N*2
pasv_ max_ port	0		

4) Configure a list of users who are allowed or prohibited from accessing the FTP server (can be skipped)

```
# vim /etc/vsftpd/user_ list
```

- When/etc/vsftpd/vsftpdconf is configured as follows,/etc/vsftpd/user is prohibited_ All users in the list access the FTP server.

```
userlist_ enable=YES
userlist_ Deny=YES (default to YES)
```

- When configured in/etc/vsftpd/vsftpdconf as follows,/etc/vsftpd/user is allowed_ All users in the list access the FTP server.

```
userlist_ enable=YES
userlist_ deny=NO
```

5) Configure a list of users who are prohibited from accessing the FTP server (can be skipped)

```
# vim /etc/vsftpd/ftpusers
```

6) Turn off the SELINUX function or change its configuration (choose one of two options)

- Turn off SELINUX function

```
# vim /etc/selinux/config

# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
```

```
SELINUX=disable
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

Restart or execute

```
# setenforce 0
```

- Change SELINUX configuration

```
# setsebool ftp_ home_ dir 1
```

be careful

When accessing the FTP server with a browser and encountering '500 OOPS: cannot change directory:/home/...', this issue may occur.

7) Turn off or configure firewall

- Turn off firewall

  Stop firewall service

```
# service iptables stop
Iptables: Clear firewall rules: [OK]
Iptables: Set the chain to policy ACCESS: filter [OK]
Iptables: Uninstalling module: [OK]
```

Check if the firewall automatically starts when turned on

```
# chkconfig --list iptables
Iptables 0: Off 1: Off 2: On 3: On 4: On 5: On 6: Off
```

Prohibit firewall from automatically starting when turned on

```
# chkconfig iptables off
```

Or

```
# chkconfig iptables off --level 2345
```

After setting, the firewall will automatically start when turned on

```
# chkconfig --list iptables
Iptables 0: Off 1: Off 2: Off 3: Off 4: Off 5: Off 6: Off
```

- Configure firewall

Set default rules

```
#Iptables - A Input - j DROP (Note: Adding this rule will block unprocessed incoming packets. If an allow rule is not added before this rule, it will block remote connections)
# iptables -A FORWARD -j ACCEPT
```

Open FTP port

```
# iptables -I INPUT -p tcp --dport 21 -j ACCEPT
# iptables -I OUTPUT -p tcp --sport 21 -j ACCEPT
# iptables -I INPUT -p tcp --dport 20 -j ACCEPT
# iptables -I OUTPUT -p tcp --sport 20 -j ACCEPT
# iptables -I INPUT -p tcp --dport 20001:21000 -j ACCEPT
# iptables -I OUTPUT -p tcp --sport 20001:21000 -j ACCEPT
```

Save firewall settings

```
# iptables-save > /etc/sysconfig/iptables
```

8) Start the vsftpd service and set it as the boot option

```
# service vsftpd start
Start vsftpd for vsftpd: [OK]

# chkconfig vsftpd on
```

9) Copy files to FTP directory

- If local is not set_ When root=/var/ftp/pub, copy the file to/home/xxxx (the user's home directory)
- If local has been set_ When root=/var/ftp/pub, copy the file to/var/ftp/pub
- If anonymous has been set_ When enable=YES, copy files to/var/ftp or/var/ftp/pub (anonymous login's home directory)

5.2.1.1.2 HTTP Server Configuration

Using Apache to Build an HTTP File Server

1) Install apr and httpd

```
# rpm -ivh
apr-1.3.9-3.el6_ 1.2.x86_ 64.rpm
apr-util-1.3.9-3.el6_ 0.1.x86_ 64.rpm apr-util-ldap-1.3.9-3.el6_ 0.1.x86_ 64.rpm
# rpm -ivh
httpd-2.2.15-15.el6.x86_ 64.rpm
```

```
httpd-manual-2.2.15-15.el6.noarch.rpm    httpd-tools-2.2.15-15.el6.x86_ 64.rpm
```

2) Modify the default configuration of the HTTP server

```
# vim /etc/httpd/conf/httpd.conf
```

Modify Server Name

```
# If your host doesn't have a registered DNS name, enter its IP address here.
# You will have to access it by its address anyway, and this will make
# redirections work in a sensible way.
#ServerName www.example.com:80
ServerName 192.168.10.114:80
```

Modify the following location and change '/var/www/html' to '/var/www/files'

You can also directly use '/var/www/html' as the file storage location and skip this step

```
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
#DocumentRoot "/var/www/html"
DocumentRoot "/var/www/files"


#
# This should be changed to whatever you set DocumentRoot to.
#
#<Directory "/var/www/html">
<Directory "/var/www/files">
```

Modify other parameters

```
#Whether to use memory mapping, default value on, set to off when mounting the
nfs system
# EnableMMAP off


#Whether to use sendfile system call, default value on, set to off when mounting
nfs system
# EnableSendfile off
```

```
#Connection timeout, default value is 60
# Timeout 60
```

Disable long file name truncation and add the following configuration.

```
<IfModule autoindex_ module>
   IndexOptions NameWidth=*
</IfModule>
perhaps
IndexOptions FancyIndexing VersionSort NameWidth*
```

be careful

If long file name truncation is not disabled, Apache will return incomplete file names, which can cause errors when loading HTTP files using wildcards.

3) Edit default welcome page configuration

```
# vim /etc/httpd/conf.d/welcome.conf
```

Comment out the following lines (by default, if there is no default page in HTML, a 403 error page will be displayed)

```
#<LocationMatch "^/+$">
# Options -Indexes
# ErrorDocument 403 /error/noindex.html
#</LocationMatch>
```

4) Turn off or configure firewall

- Turn off firewall

  Stop firewall service

  ```
  # service iptables stop
  Iptables: Clear firewall rules: [OK]
  Iptables: Set the chain to policy ACCESS: filter [OK]
  Iptables: Uninstalling module: [OK]
  ```

  Check if the firewall automatically starts when turned on

  ```
  # chkconfig --list iptables
  Iptables 0: Off 1: Off 2: On 3: On 4: On 5: On 6: Off
  ```

  Prohibit firewall from automatically starting when turned on

```
# chkconfig iptables off
```

Or

```
# chkconfig iptables off --level 2345
```

After setting, the firewall will automatically start when turned on

```
# chkconfig --list iptables
Iptables 0: Off 1: Off 2: Off 3: Off 4: Off 5: Off 6: Off
```

- Configure firewall

  Set default rules

```
# iptables -A INPUT -j DROP
# iptables -A FORWARD -j ACCEPT
```

  Open HTTP port

```
# iptables -I INPUT -p tcp -m tcp --dport 80 -j ACCEPT
# iptables -I OUTPUT -p tcp -m tcp --sport 80 -j ACCEPT
```

  Save firewall settings

```
# iptables-save > /etc/sysconfig/iptables
```

5) Start the httpd service and set it as the boot option

```
# service httpd start
Starting httpd: [OK]


# chkconfig httpd on
```

6) Copy the data file to/var/www/files (or/var/www/html)

7) Accessing with a browser http://192.168.10.114 You can see the file list (previously configured ServerName 192.168.10.114:80)

5.2.1.1.3 HDFS Server Configuration

Building an HDFS server using Apache Hadoop 2.6.0

1) Preparation of Hadoop Cluster Environment

   Operating system user: gbase

   The SSH mutual trust between nodes in the cluster has been established.

   The cluster has been configured with C3 tools.

   Open source product version:

Apache Hadoop 2.6.0

JVM version 1.6 or 1.7

Example:

Table -5145 Cluster Node Function Planning

IP	host name	function
192.168.10.114	ch-10-114	NameNode, DataNode
192.168.10.115	ch-10-115	DataNode
192.168.10.116	ch-10-116	DataNode

2) Host Name Configuration

The host names of each node need to be configured correctly, as shown in the 192.168.10.114 node example. Other nodes can directly copy this configuration.

```
127.0.0.1      localhost localhost.localdomain localhost4 localhost4.localdomain4
::1            localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.10.114 ch-10-114
192.168.10.115 ch-10-115
192.168.10.116 ch-10-116
```

be careful

If the first line is configured in the following form, it is an error. After installation, there will be a situation where Hadoop's Datanode cannot connect to Namenode.

```
127.0.0.1      ch-10-114 localhost localhost.localdomain localhost4
localhost4.localdomain4
```

If there are no DNS servers in the cluster that can resolve the host names of Hadoop's Namenodes and Datanodes, then it is necessary to configure the/etc/hosts file on each coordinator node that performs the load task, as well as on each data node in the cluster, to include the IP addresses and host name mappings of Hadoop's Namenodes and Datanodes mentioned above. If the/etc/hosts file is not configured, an error similar to "Couldn't resolve host name" will be reported when loading files on the HDFS server.

Inspection method:

- Through jps inspection, it was found that the DataNode had started, but upon checking the logs on the DataNode, it was found that the DataNode was constantly trying to connect to the 9000 port (RPC port of HDFS) of the NameNode node.
- Execute netstat - an on the NameNode node and see the following information:

```
$ netstat -an | grep 9000
tcp    0    0 127.0.0.1:9000        0.0.0.0:*               LISTEN
```

Error reason: The TCP listening IP is 127.0.0.1, which results in only the local machine being able to connect to port 9000. The reason is that the/etc/hosts file configuration of NameNode is incorrect.

Solution: Remove the red font (ch-10-114) from the first line, or place the content of the first line after it.

```
192.168.10.114 ch-10-114
192.168.10.115 ch-10-115
192.168.10.116 ch-10-116
127.0.0.1      localhost localhost.localdomain localhost4 localhost4.localdomain4
::1            localhost localhost.localdomain localhost6 localhost6.localdomain6
```

Restart HDFS and use netstat - an | grep 9000 again to check that the port and IP are correct.

```
$ netstat -an | grep 9000
tcp    0    0 192.168.10.114:9000        0.0.0.0:*               LISTEN
```

3) Catalog Planning

Table -5146 Catalog Planning

catalogue	purpose
/home/gbase/bin	Place the Hadoop ecosystem, including Hadoop and others
/home/gbase/hdfs	Place HDFS files, including tmp, name, and data

Add environment variable ${HADOOP_HOME}

```
$ echo “export HADOOP_HOME=/home/gbase/bin/Hadoop-2.6.0”>> ~/.bashrc
$. ~/. bashrc
```

be careful

The ${HADOOP_HOME} in the following text refers to/home/gbase/bin/Hadoop-2.6.0

4) Preparing Hadoop 2.6.0

Extract hadoop-2.6.0.tar.gz to the/home/gbase/bin of each node.

```
$ tar xfz hadoop-2.6.0.tar.gz -C /home/gbase/bin
```

5) Configure hadoop env.sh

File path:

```
${HADOOP_HOME}/etc/hadoop/hadoop-env.sh
```

```
$ cd ${HADOOP_HOME}
$ vi etc/hadoop/hadoop-env.sh
```

Both Name node and Data node are configured as follows.

Export JAVA_ HOME=$JAVA_ Modify HOME to:

```
export JAVA_ HOME=/usr/lib/jvm/jre-1.6.0-openjdk.x86_ sixty-four
```

Export HADOOP_ CONF_ DIR=${HADOOP_CONF_DIR: - "/etc/hadoop"} Modify to:

```
export HADOOP_ CONF_ DIR=/home/gbase/bin/hadoop-2.6.0/etc/hadoop
```

6) Configure the core site. xml file

File path: ${HADOOP_HOME}/etc/hadoop/core site. xml

```
$ cd ${HADOOP_HOME}
$ vi etc/hadoop/core-site.xml
```

Both Name node and Data node are configured as follows

```
<configuration>
    <property>
        <name>fs.default.name</name>
        <value> hdfs://ch-10-114:9000 </value>
    </property>
    <property>
        <name>hadoop.tmp.dir</name>
        <value>file:/home/gbase/hdfs/tmp</value>
    </property>
</configuration>
```

7) Configure hdfs site.xml

File path:

```
${HADOOP_HOME}/etc/hadoop/hdfs-site.xml
```

```
$ cd ${HADOOP_HOME}
$ vi etc/hadoop/hdfs-site.xml
```

Name node configuration

```
<configuration>
```

```
    <property>
        <name>dfs.replication</name>
        <value>2</value>
    </property>
    <property>
        <name>dfs.name.dir</name>
        <value>file:/home/gbase/hdfs/name</value>
        <description>name node dir </description>
    </property>
    <property>
        <name>dfs.permissions</name>
        <value>false</value>
    </property>
</configuration>
```

Data Node Configuration

```
<configuration>
    <property>
        <name>dfs.data.dir</name>
        <value>file:/home/gbase/hdfs/data</value>
        <description>data node dir</description>
    </property>
</configuration>
```

8) Configure Masters and Slaves

File path:

```
${HADOOP_HOME}/etc/hadoop/masters
${HADOOP_HOME}/etc/hadoop/slaves
```

Simply configure it on the NameNode node.

```
$ cd ${HADOOP_HOME}
$ vi etc/hadoop/masters
```

HADOOP_HOME}/etc/hadoop/masters file content

```
ch-10-114
```

```
$ cd ${HADOOP_HOME}
$ vi etc/hadoop/slaves
```

HADOOP_HOME}/etc/hadoop/slides file content

```
ch-10-114
ch-10-115
ch-10-116
```

9) Format NameNode

The formatting of NameNode needs to be done before starting HDFS.

```
$ cexec rm -fr /home/gbase/hdfs/*
$ cd ${HADOOP_HOME}
$ bin/hdfs namenode -format
```

10) Start HDFS

```
$ cd ${HADOOP_HOME}
$ sbin/start-dfs.sh
```

After startup, check the processes of each node through jps, and the following is the correct startup.

```
$ cexec jps
************************ test ************************
--------- 192.168.10.114---------
31318 SecondaryNameNode
31133 NameNode
31554 Jps
--------- 192.168.10.115---------
10835 DataNode
11000 Jps
--------- 192.168.10.116---------
10145 DataNode
10317 Jps
```

11) Stop HDFS

```
$ cd ${HADOOP_HOME}
$ sbin/stop-dfs.sh
```

5.2.1.1.4 SFTP Server Configuration

The SFTP service does not require the deployment of additional software packages, just enable the sshd service.

1) Check if the sshd service has been started

```
# service sshd status
openssh-daemon (pid  2243) is running...
```

2) Default configured directory access. When using the default configuration, sshd does not restrict user directory access. After logging in through sftp, users can jump between any directory with access permissions. In this case, the file path in the URL of the following load statement is the absolute path of the system:

```
load data infile ‘ sftp://gbase:gbase @192.168.10.114/opt/data/test.tbl’into table test.t data_
format 3;
```

3) Modify the sshd default configuration.

When the number of concurrent loading tasks and the maximum number of single task loading machines in the cluster are large, there may be a failure to load the sftp file. At this time, the sshd configuration file can be modified as follows.

Edit/etc/ssh/sshd_ Config file

```
# vi /etc/ssh/sshd_ config
```

Modify the configuration file with the following bold font content

```
#The value of MaxStartups is represented as "start: rate: full", with a default value of 10:30:100.
When the number of unauthenticated connections reaches start (10), new connection attempts
with a "rate/100" (30%) may be rejected by sshd. When the number of unauthenticated
connections reaches full (100), all new connection attempts will be rejected.
#When the maximum number of concurrent loading tasks in the cluster is N and the maximum
number of single loading tasks (max_data_processors) is M, the recommended value for
MaxStartups is M * N+10:30: M * N * 2
#
MaxStartups 10:30:100MaxStartups 20:30:100
```

For security reasons, it is desired to restrict the access permissions of SFTP login users and only allow them to be active in their home directory.

Enabling the sshd directory locking function requires the use of chroot. After openssh 4.8p1, chroot is supported. The following command can be used to check the current system's openssh version.

```
# ssh -V
OpenSSH_ 5.3p1, OpenSSL 1.0.0-fips 29 Mar 2010
```

Edit/etc/ssh/sshd_ Config file

```
# vi /etc/ssh/sshd_ config
```

Modify the configuration file with the following bold font content

```
# override default of no subsystems
```

```
#Modify the default subsystem to internal sftp
#Subsystem      sftp    /usr/libexec/openssh/sftp-server
Subsystem       sftp    internal-sftp

# Example of overriding settings on a per-user basis
#Match User sftp means that the following rules only match users with the name sftp. If multiple user names need to be matched, they should be separated by commas. Match Group sftp can also be used to match groups with the name sftp. Similarly, if multiple groups need to be matched, multiple group names should be separated by commas
Match User sftp
#       X11Forwarding no
#       AllowTcpForwarding no
#Force the execution of the in-process sftp server, ignoring commands in the~/. ssh/rc file
ForceCommand internal-sftp
#Use chroot to specify the user's root directory to% h, where% h represents the user's home directory. Optional parameters include% u, which represents the username
ChrootDirectory %h
```

be careful

Key points for configuring permissions for the sftp directory:

- The directory owner from the directory specified by ChrootDirectory up to the system root directory can only be root.
- From the directory specified by ChrootDirectory up to the system root directory, there is no group write permission, that is, the permission value cannot exceed 755.

4) Restart the sshd service after configuration is complete

```
# service sshd restart
```

With directory locking configured, the file path in the URL of the following load statement is the relative path of the system, and the absolute path of test.tbl should be/home/gbase/opt/data/test.tbl

```
load data infile ‘ sftp://gbase:gbase @192.168.10.114/opt/data/test.tbl’into table test.t data_ format 3;
```

5.2.1.1.5 Deployment and Use of GBFS Dedicated File Server

GBFS dedicated file server is a binary executable program specifically designed for

loading GBase 8a MPP Cluster database data.

Usually provided to users in the form of a gbfs-9.5.3.22-redhat7.3.tar.bz2 file package, users only need to use the following command to extract the compressed package and run it.

Taking gbfs-9.5.3.22-redhat7.3.tar.bz2 as an example for explanation:

tar xvf gbfs-9.5.3.22-redhat7.3.tar.bz2

After the decompression is completed, a gbfs folder will be generated in the current directory, which includes the gbfs main program and BUILDINFO (compilation information). Use gbfs -? The command may view help information for the gbfs program.

[root@rhel73-1 gbfs]# ./ gbfs -?

./gbfs ver 9.5.3.22.126635 for unknown-linux-gnu on x86_ sixty-four

Copyright 2004-2021 General Data Technology Co.Ltd.

GBase File Server

Usage: ./ gbfs [OPTIONS]

-V, --version Get version info.

-?, -- help Get help info.

-P, --port Port number to use for connection or 6666 for default,

valid range: [1025,65535] order of preference.

-H, --home-dir The GBase file server home dir, default: current user home dir.

-L, --log-dir The GBase file server logs dir, default: /tmp/.

The help information includes version information of gbfs and an introduction to usage methods. The parameters are introduced as follows:

-P&-- port is the port number that the GBFS dedicated server listens to when working. The default is 6666.

-H&-- home dir is the HOME directory where gbfs works, similar to the HOME directory function of FTP. By default, it is the HOME directory of the current boot user. This parameter is mainly used to support the relative path function of gbfs.

For example:

Run as a GBase user. So the default HOME directory for gbfs is:/home/gbase/, if user data is stored in/home/gbase/data/. Users may directly load files using the following URL.

gbfs://192.168.146.20/data/test.tbl

The URL of the absolute path relative to it is as follows:

gbfs://192.168.146.20//home/gbase/data/test.tbl

Users can configure this parameter based on actual scenarios.

-L&-- log dir is the log file storage directory of gbfs. After gbfs starts, a new gbfs will be created in this directory_ port.log。 The default is in the/tmp/directory.

It is usually recommended to place the gbfs dedicated file server in the background for operation:

[gbase@rhel73-1 gbfs]$ ./ gbfs &

[1] 23302

[gbase@rhel73-1 gbfs]$ IPv6 is available.

gbfs is ready for connections. home dir:/home/gbase/, log dir:/tmp/, port:6666.

5.2.1.2 Load status monitoring

Function Description

After the loading task is started, the status information of the current loading task can be viewed through SQL.

Grammar format

```
SELECT * FROM information_ schema.load_ status;
```

The status information of all running load tasks is recorded in the status information table in Figure -51.

Field	Type	Null	Key	Default	Extra
SCN	bigint(20)	NO		0	
DB_NAME	varchar(64)	NO			
TB_NAME	varchar(64)	NO			
IP	varchar(20)	NO			
STATE	varchar(20)	NO			
START_TIME	datetime	NO		0000-00-00 00:00:00	
ELAPSED_TIME	bigint(20)	NO		0	
AVG_SPEED	bigint(20)	NO		0	
PROGRESS	bigint(8)	NO		0	
TOTAL_SIZE	bigint(20)	NO		0	
LOADED_SIZE	bigint(20)	NO		0	
LOADED_RECORDS	bigint(20)	NO		0	
SKIPPED_RECORDS	bigint(20)	NO		0	
DATA_SOURCE	varchar(1024)	NO			
SQL_CMD	varchar(4096)	NO			

Table -5147 Definition of Fields in the Memory Table

Field Name	Explanation of Meaning
Field	Description
SCN	SCN number
DB_ NAME	Database name
TB_ NAME	Table Name
IP	Loader IP
STATE	Load Status

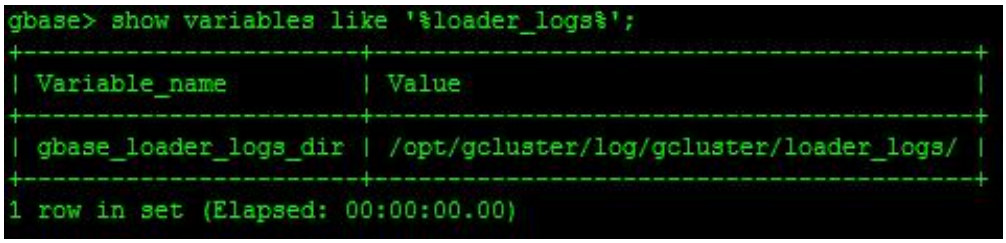
Field Name	Explanation of Meaning
START_ TIME	Load start time
ELAPSED_ TIME	Load End Time
AVG_ SPEED	Loading speed
PROGRESS	Loading progress
TOTAL_ SIZE	Total file length
LOADED_ SIZE	Amount of data loaded
LOADED_ RECORDS	Number of loaded data entries
SKIPPED_ RECORDS	Skip Number of Data Entries
DATA_ SOURCE	data source
SQL_ CMD	SQL for loading tasks

5.2.1.3 Load Log Summary and Query

The log summary and query function summarizes the error data logs and traceability information logs loaded at once to the loading initiation node, and provides corresponding queries and retrieval of logs. This function depends on the activation of the GNS function in the table.

By variable gbase_ loader_ logs_ Dir specifies the log file summary path, which defaults to the loader in the gcluster log directory of the loading initiator node ($GCLUSTER_HOME/log/gcluster/)_ Log directory, and create a TASK file under this path for this TASK_ The sub folder named by ID will store the summary logs in that sub folder. When loading is complete, create a TASK_ ID_ loader_ A log named result.log, and write the result information of this load to the log file. This variable supports both set mode modification and configuration file mode modification.

Figure 52 Summary Path of Log Files



By variable gbase_ loader_ logs_ The switch for the collection change control cluster loading log summary function, with a valid value of [0,1] and a default value of 1, indicates that the summary function is enabled. This variable supports both set and configuration file modifications.

For cluster loading, if gbase_ loader_ logs_ Collect is 1, error data and traceability information are summarized to the loading initiation node and stored in GBase_ loader_ logs_ Dir specifies the directory, otherwise error data and traceability information logs will not be summarized.

be careful

- For the naming of logs, the summary function does not change the file name of the logs and follows existing naming rules;
- For the number of logs, the summary function does not merge the log files, that is, the summary function only summarizes the log files generated by each data loading node to the loading initiation node.

As shown in Figure 53

```
[gbase@RH_6_72 ~]$ ll /opt/gcluster/log/gcluster/loader_logs/
total 8
drwxrwxr-x 2 gbase gbase 4096 Dec 30 10:07 131075
drwxrwxr-x 2 gbase gbase 4096 Dec 30 10:08 131076
[gbase@RH_6_72 ~]$ ll /opt/gcluster/log/gcluster/loader_logs/131076
total 1280376
-rw-rw---- 1 gbase gbase 189965464 Dec 30 10:07 131076_test_lineitem_n1_192.168.6.72_20161230100740.err
-rw-rw---- 1 gbase gbase 137734542 Dec 30 10:07 131076_test_lineitem_n1_192.168.6.72_20161230100740.trc
-rw-rw---- 1 gbase gbase 189965260 Dec 30 10:07 131076_test_lineitem_n1_192.168.6.73_20161230100740.err
-rw-rw---- 1 gbase gbase 137825915 Dec 30 10:07 131076_test_lineitem_n1_192.168.6.73_20161230100740.trc
-rw-rw---- 1 gbase gbase 189965369 Dec 30 10:07 131076_test_lineitem_n2_192.168.6.74_20161230100740.err
-rw-rw---- 1 gbase gbase 137813242 Dec 30 10:07 131076_test_lineitem_n2_192.168.6.74_20161230100740.trc
-rw-rw---- 1 gbase gbase 189965273 Dec 30 10:07 131076_test_lineitem_n3_192.168.6.75_20161230100044.err
-rw-rw---- 1 gbase gbase 137850597 Dec 30 10:07 131076_test_lineitem_n3_192.168.6.75_20161230100044.trc
[gbase@RH_6_72 ~]$
```

5.2.1.3.1 Loading Error Data and Tracing Information Retrieval

Function Description

Used for error data and traceability information retrieval.

Grammar format

SHOW [GCLUSTER] LOAD LOGS *task_ id* LIMIT {[*offset,] row_count*};

Table 5148 Parameter Description

Field Name	Explanation of Meaning
GCLUSTER	Optional parameter, add this parameter to display information on all coordinator nodes.
Task_ id	The process identifier of the load task to trace.
offset	Start of information.
Row_ count	Number of records.

Table 5149 Query Result Information Table Definition

Field Name	Explanation of Meaning
TASK_ ID	Load ID
DB_ NAME	Load library name
TB_ NAME	Load Table Name
ERR_ DATA_ IP	Node IP that generated incorrect data
FILE_ NAME	Load File Name

Field Name	Explanation of Meaning
FILE_ OFFSET	Error data offset
RECORD_ LEN	Incorrect Data President
ERR_ COLUMN	Error data column number
ERR_ REASON	Specific reasons for incorrect data
ERR_ DATA	Incorrect data

explain

- The show syntax query defaults to returning 10 error data and traceability information with an offset of 0 to length 10. If you want to query more data, you can adjust the offset and length.
- Show syntax queries the error data and traceability information of the current coordinator node for retrieval, using show load logs task_ id limit offset, row_ Count to query and return row_ Count query results.
- Query all coordinator node error data and traceability information for retrieval, using show gcluster load logs task_ id limit offset, row_ Count to query and return row_ Count query results.
- ERR_ The length of DATA is defined as 4096 bytes. It can cover the vast majority of scenarios. For error data exceeding the length, truncation processing is performed during display, and 4096 bytes are actually read.
- The show query function can only query the loading error data and traceability information in the current summary directory. If the user is interested in GBase_ loader_ logs_ After making changes to dir, the data in the originally specified directory will not be queried.
- Show syntax adds user query permission control function. By default, only the error data and traceability information of the current user's specified loading task can be queried. Users with process permission can query the error data and traceability information of other users' specified loading tasks.

For example:

Show load logs 100 Show tasks_ Top 10 error data messages for task id 100

Show load logs 100 limit 5 display tasks_ The first 5 error data messages for task id 100

Show load logs 100 limit 0,5 Show tasks_ The first 5 error data messages for task id 100

Show load logs 100 limit 1,5 Display tasks_ The following 5 error data messages for task id 100 starting from item 1

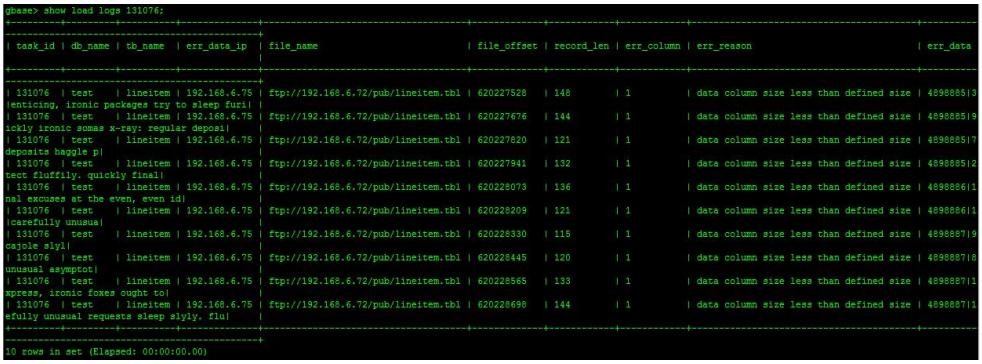
Show gcluster load logs 101 Show tasks on all coordinator nodes_ Top 10 error data messages for task ID 101

Example

- Example 1: Query task_ The first 10 error data and traceability information loaded with ID 131076 times.

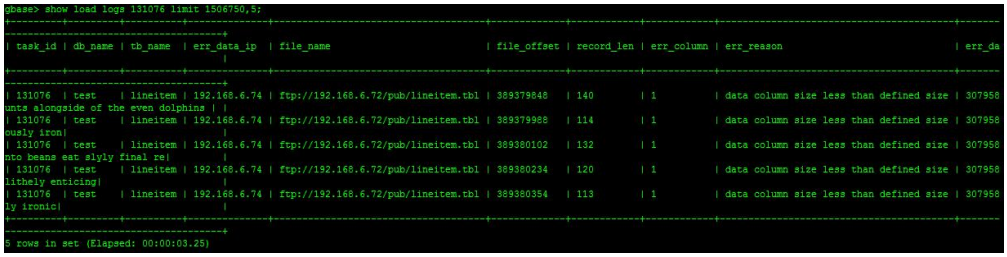
```
show load logs 131076;
```

As shown in Figure 54



- Example 2: Query task_ Information related to the last 5 erroneous data starting from 1506750 loaded with ID 131076.

As shown in Figure 55



5.2.1.3.2 Load result information statistics log

Write the loading result information to the log file loader when the loading is completed_ In result.log, the loading result information is a regular text file stored with '|' as the column separator and ' n' as the row separator. It is stored in the log directory of the initiating node gccluster ($GCLUSTER_HOME/log/gcluster/) and does not support specifying a storage path.

Table 5150 Field Specific Meaning:

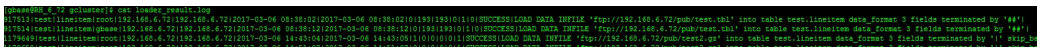
Field Name	Explanation of Meaning
TASK_ ID	Load ID
DB_ NAME	Load Database Name
TB_ NAME	Load Table Name
USER	Current loaded username
ACCESS_ IP	Load origin IP
HOST_ IP	Client IP
START_ TIME	Load start time
END_ TIME	Load End Time

Field Name	Explanation of Meaning
ELAPSED_ TIME	Loading time
TOTAL_ SIZE	Total size of loaded files
AVERAGE_ SPEED	Average loading speed
LOADED_ RECORDS	Number of loaded data entries
SKIPPED_ RECORDS	Skipped number of loading data
IGNORED_ FILES	Number of skipped files loaded
RESULT	Load Results
SQL_ CMD	Load SQL
MESSAGE	error message

be careful

SQL_ The inclusion of ' n' in CMD and MESSAGE is replaced by spaces in the log file.

As shown in Figure -56



5.2.1.3.3 Upload error data and traceability information logs directly to the FTP/SFTP server

Parameter global gbase_ loader_ logs_ Collect is used to control whether gnode uploads error data and traceability information logs directly to ftp/sftp during the loading process. After loading is completed, gcluster uploads the loading result information directly to ftp/sftp.

Parameter global gbase_ loader_ logs_ Collect is used to control whether the log summary function is enabled. The default value is 1, which means it is enabled.

- When the default control log summary function is enabled, the upload path for loading log files follows the global gbase_ loader_ logs_ The path specified by the dir parameter.

```
set global gbase_ loader_ logs_ dir=' ftp://gbase:gbase @192.168.6.15/loadlogs '
//Summary directory of error data and traceability log files
load data infile ' ftp://gbase:gbase @127.0.0.1/data/a.tbl' into table customer;
```

- When the control log summary function is turned off, the upload path follows the trace_ The path specified by path.

```
set global gbase_ loader_ logs_ collect=0;    // Turn off the control log summary function
```

```
load data infile ' ftp://gbase:gbase @127.0.0.1/data/a.tbl' into table customer trace_ path ' ftp://gbase:gbase @192.168.6.15/loadlogs';
```

be careful

Set gbase_ loader_ logs_ After the dir variable is set to the FTP/SFTP directory URL, the error data and traceability information logs will be directly uploaded to FTP/SFTP. Since they are not saved on the initiating node GCLUSTER, using show load logs will not retrieve the results, and export load logs cannot be used to export log files.

5.2.1.3.4 Load Result Information Memory Table Query

Load result information through information_ Load in the schema library_ RESUL T and CLUSTER_ LOAD_ Query the RESULT table,

As shown in Figure 57

```
gbase> show tables like '%LOAD_RESULT%';
+-----------------------------------------------+
| Tables_in_information_schema (%LOAD_RESULT%) |
+-----------------------------------------------+
| LOAD_RESULT                                   |
| CLUSTER_LOAD_RESULT                           |
+-----------------------------------------------+
2 rows in set (Elapsed: 00:00:00.00)
```

The definition of the loading result information table is consistent with the definition of the loading result log column.

As shown in Figure 58

```
[gbase@RH_6_72 gcluster]$ gccli -ugbase -pgbase20110531

GBase client 8.6.1.1 build 72003. Copyright (c) 2004-2017, GBase.  All Rights Reserved.

gbase> desc information_schema.load_result;
+-----------------+---------------+------+-----+---------------------+-------+
| Field           | Type          | Null | Key | Default             | Extra |
+-----------------+---------------+------+-----+---------------------+-------+
| TASK_ID         | bigint(20)    | NO   |     | 0                   |       |
| DB_NAME         | varchar(64)   | NO   |     |                     |       |
| TB_NAME         | varchar(64)   | NO   |     |                     |       |
| USER            | varchar(64)   | NO   |     |                     |       |
| ACCESS_IP       | varchar(20)   | NO   |     |                     |       |
| HOST_IP         | varchar(20)   | NO   |     |                     |       |
| START_TIME      | datetime      | NO   |     | 0000-00-00 00:00:00 |       |
| END_TIME        | datetime      | NO   |     | 0000-00-00 00:00:00 |       |
| ELAPSED_TIME    | bigint(20)    | NO   |     | 0                   |       |
| TOTAL_SIZE      | bigint(20)    | NO   |     | 0                   |       |
| AVERAGE_SPEED   | bigint(20)    | NO   |     | 0                   |       |
| LOADED_RECORDS  | bigint(20)    | NO   |     | 0                   |       |
| SKIPPED_RECORDS | bigint(20)    | NO   |     | 0                   |       |
| IGNORED_FILES   | bigint(20)    | NO   |     | 0                   |       |
| RESULT          | varchar(20)   | NO   |     |                     |       |
| SQL_CMD         | varchar(4096) | NO   |     |                     |       |
| MESSAGE         | varchar(1024) | NO   |     |                     |       |
+-----------------+---------------+------+-----+---------------------+-------+
17 rows in set (Elapsed: 00:00:00.01)
```

be careful

- Support select query format, query loading result information
- Only query the current coordinator node, select the query form, query the loading information, and the table name is: LOAD_ RESULT Example:

  select * from information_ schema.load_ result;
- Query all coordinator nodes, select query format, query loading information, and table name: CUSTER_ LOAD_ RESULT

  For example:

  select * from information_ schema.cluster_ load_ result;
- The loading result information query function implements user permission control. For users with PROCESS permission, they can query the information that has been loaded by all users in the current cluster. For users without this permission, they can only query their own loading result information.

5.2.1.3.5 Load skip file list log echo

Enable SKIP during loading process_ BAD_ The FILE function, when this parameter is set to 1, if gbased finds any skip error file information, it will send the skip error data file information back to the loading initiation node and write it to the gcluster's express log file.

This function depends on the activation of the GNS function. If the GNS function is turned off, that is, gbase_ gns_ share_ When connection=0 and GBase detects a skip error file, the skip error data file information is written to the GBase's e

xpress log file, consistent with the existing loading behavior.

5.2.2 Cluster batch loading statements

5.2.2.1 Cluster loading syntax

Grammar format

```
LOAD DATA INFILE 'file_ list'
INTO TABLE [[vc_name.]database_ name.] table_ name
[options]
options:
   [CHARACTER SET charset_name]
  [DATA_FORMAT number [HAVING LINES SEPARATOR]]
  [NULL_VALUE 'string']
  [FIELDS
     [TERMINATED BY 'string']
     [ENCLOSED BY 'string']
      [PRESERVE [LEADING | TRAILING] BLANKS]
     [AUTOFILL]
      [LENGTH 'string']
     [TABLE_FIELDS 'string']
  ]
  [LINES
     [TERMINATED BY 'string']
  ]
  [MAX_BAD_RECORDS number]
  [DATETIME FORMAT format]
  [DATE FORMAT format]
  [TIMESTAMP FORMAT format]
  [TIME FORMAT format]
  [TRACE number]
  [TRACE_PATH    'string']
  [NOSPLIT]
  [PARALLEL number]
  [MAX_DATA_PROCESSORS number]
  [MIN_CHUNK_SIZE number]
  [SKIP_BAD_FILE number]
   [SET col_name = value[,...]]
  [IGNORE NUM LINES]
   [FILE_FORMAT format]
```

Parameter Description

- FILE_ LIST: List of files to be loaded, or the directory where the data files to be loaded are located. Supports specifying data file paths through URLs, using commas (',') as separators for multiple files/directories.

```
scheme://host:port/path , scheme://host:port/path
```

Supports loading of the following data sources:

Commonly used data sources such as S3, local, https, FTP, sftp, hdfs, gbfs, kafka, etc.

---

be careful

The list of files to be loaded in SQL supports loading files from different loading sources together, such as:

' ftp://192.168.0.1/pub/lineitem.tbl , http://192.168.0.2/lineitem.tbl '

However, local files and other types of loading sources cannot appear simultaneously, otherwise an error message 'mix use file protocol' will be reported.

Supports loading Amazon S3 data and supports the following four URL formats:

1) s3n://AWSAccessKeyId: AWSSecretKey@s3-aws-region.amazonaws.com /region/bucket/key
2) s3ns://AWSAccessKeyId: AWSSecretKey@s3-aws-region.amazonaws.com /region/bucket/key
3) s3v://AWSAccessKeyId: AWSSecretKey@bucket.s3-aws-region.amazonaws.com /region/key
4) s3vs://AWSAccessKeyId: AWSSecretKey@bucket.s3-aws-region.amazonaws.com /region/key

explain

- S3n, s3ns, s3v, and s3vs are custom S3 protocol prefixes that are not case sensitive and have the following meanings:
  1) S3n represents the use of HTTP protocol and path type URLs to access S3 server terminal nodes;
  2) S3ns represents the use of HTTPS protocol and path type URLs to access S3 server terminal nodes;
  3) S3v represents the use of HTTP protocol and virtual hosting type URLs to access S3

server terminal nodes;

4) S3vs represents the use of HTTPS protocol and virtual hosting type URLs to access S3 server terminal nodes;

- AWSAccessKeyId is the AWS account access key ID, which is a 20 character alphanumeric string. For example: AKIAIOSFODNN7EXAMPLE.
- AWSSecretKey is the AWS account secret access key, which is a 40 character string. For example: wJalrXUtnFEMixK7MDENGxbPxRfiCYEXAMPLEKEY.
- S3-aws-region.amazonaws.com is the endpoint of S3.
- Region is the name of the region where the storage bucket belongs.
- Bucket is the storage bucket name (bucket).
- Key is the key name (key) of the object in the bucket.

Support local data source loading:

1) Supports specifying local files on one or more data nodes for loading. use file://host +abs_ Path, multiple file://host +abs_ The paths are separated by commas and support direct read mode to load local files of specified cluster data nodes.
2) Support the simultaneous loading of files on each node by specifying all data nodes. Using file://+abs_ Path, multiple files://+abs_ The paths are separated by commas and support direct read mode to load local files of all data nodes in the cluster.

be careful

Local loading under multiple instances does not support file://+abs_ Path syntax, needs to be rewritten as file://host +abs_ The syntax form of path.

FILE_ The file name and directory sections in LIST support the use of wildcards, as shown in the following table. By default, matching paths and files is performed. When disabling directory and file wildcard functions, wildcard characters appearing in SQL are treated as special characters according to the usage constraints in the "Usage Constraints" section.

Table 5151 Wildcard Description

Coordination character	meaning	explain
*	Match 0 or more characters	a*b There can be any character of any length or none between a and b, such as aabcb, axyzb,

		a012b, ab.
?	Match any character	a? b There must and can only be one character between a and b, which can be any character, such as aab, abb, acb, a0b.
[list]	Match any single character in the list	a[xyz]b There must and can only be one character between a and b, but it can only be x, y, or z, such as axb, ayb, azb.
[!list]	Match any single character except in the list	a[!0-9]b There must and can only be one character between a and b, but they cannot be Arabic numerals, such as axb, aab, a-b.
[c1-c2]	Match any single character in c1-c2, such as: [0-9] [a-z]	a[0-9]b There must and can only be one character between 0 and 9, such as a0b, a1b a9b。
{string1, string2,...} (This feature is not supported, {} is processed as a regular character)	Matches one of the strings sring1 or string2 (or more), and string can also be a wildcard character	a{abc,xyz,123}b Only one of the three strings abc, xyz, or 123 can exist between a and b. (Match three times to obtain three results, which means there can be up to three duplicate results) {1.. 3}. txt This situation can only be 1. txt, 2. txt, or 3. txt A {1,2} b {3,4} This will sequentially match the output result as a1b3 a1b4 a2b3 a2b4

For example:

http://10.10.1.1/data/??????/ *.tbl

- CHARACTER SET: Used to specify the encoding format of the data file to be loaded, currently supporting two formats: GBK and UTF8. When omitted, it is considered that transcoding is not necessary.
- DATA_ FORM: Used to specify which method to use to parse data files and load them. Specify 3 to load using text mode. Specify 4 to indicate loading using fixed length method. If a column of data may contain row delimiters, you need to enter the 'HAVING LINES SEPARATOR' clause in SQL. When specified as 5, it indicates the use of text file loose mode, where the data source file is a text file with line breaks and bounding characters in the bounding box, or a text file with multiple columns and fewer columns. Specify 8 or orc to load the orc file.
- NULL_ Value: Used to specify null value characters, supports combinations of no more than 15 arbitrary characters, and parameter values are enclosed in quotation marks in the same way as field delimiters.
- Terminated BY: Used to specify field delimiters, supports combinations of no more than 15 arbitrary characters, supports any character, and parameter values are

enclosed in quotation marks, only valid when loaded in text mode. It can be specified in four ways: character itself (only visible characters, such as "|"), C-style escape character (such as " a"), xhh hexadecimal (such as " xFF"), or x " hexadecimal (such as "x'09 '"). For example, '|' indicates using | as the separator character.

- ENCLOSED BY: Used to specify the field delimiter, supports any single character, and parameter values are enclosed in single quotes. It is only valid when loaded in text mode. It can be specified in four ways: character itself (only visible characters, such as "|"), C-style escape character (such as " a"), xhh hexadecimal (such as " xFF"), or x " hexadecimal (such as "x'09 '").
- PRESERVE [LEADING | TRAILING] BLANKS: used to set whether to reserve the space at both ends of the field content. Optional parameters can choose to reserve the left space or the right space. By default, no space is reserved.
- AUTOFILL: used to set whether to enable the automatic filling function for missing columns. After enabling this parameter, the field data of missing delimiters is loaded according to the default or NULL values, and automatic filling is not enabled by default.
- LENGTH: The parameter used to set the field length when loading in fixed length mode. When importing fixed length format data, set the length of each field, and separate multiple fields with commas.
- TABLE_ FIELDS: Used to specify column loading, and for date time types, the format of each column can be set. For fields that do not need to be loaded in the data file during the data loading process, if this field is present in the table definition, table can be used_ The filler keyword in the fields parameter ignores the data and fills it with NULL; If the field is not present in the table definition, it can be found in the table_ Add column names that do not exist in the table to the fields parameter, and this field data will be ignored when loaded.
- SET: Load the specified column value, and the loading system will load the file to be loaded and the specified column value into the table of the cluster system. The input type should be constant, including string, integer value, floating-point value, and NULL.

  1) Supports specifying loading values for all column types;

  2) The specified column value is a constant value (including NULL), including string (surrounded by single quotation marks), decimal value (10), floating point value (10.9), NULL, string represented by hexadecimal (0xbac3), Scientific notation (10e4);

  3) Supports multiple columns specifying loading values simultaneously. The maximum number of SET table columns can be -1. If the set number of columns matches the number of columns in the table definition, an error will be reported: Specified all fields;

  4) Support format=3, format=4, and format=5;

Usage restrictions:

1) Entering values other than constant values, such as column names, expressions, etc., will result in an error message of Column 'addr' should be const value;

2) The specified column cannot exist in Table_ In FIELDS, otherwise an error will be reported;

3) If AUTOFILL is not specified, the sum of the number of columns in the specified value and the number of columns in the data must be equal to the table definition or Table_ The number of columns in FIELDS (if TABLE-FIELDS is specified), otherwise incorrect data will be generated; If AUTOFILL is specified, it can be less than the number of columns defined in the table, and missing columns will be automatically filled in. If TABLE_ The number of FIELDS columns+SET columns is less than the number of columns defined in the table, and can be loaded normally. Columns that are not involved are filled in according to the default value;

4) The same column cannot be specified repeatedly in SQL, otherwise an error will be reported.

- Terminated BY: Line separator, supports combinations of no more than 15 arbitrary characters, supports any character, and parameter values are enclosed in quotation marks. The specified method is the same as the bounding box. The default line separator is' n '.
- MAX_ BAD_ RECORDS: Set the upper limit for the number of error data rows in each loaded task. When the number of error data rows generated by this loading task is greater than max_ bad_ When the values set by records are set, the loading task rolls back and the loading tool exits with an error. Not specifying this parameter means that the number of errors is not limited. When specifying this parameter, the value range of this parameter is: [0, 4294967295]. 0 means that if there is any incorrect data, an error will be reported to exit.

  The calculation method for the maximum number of loading errors: All cluster nodes are calculated independently. Once the error data reaches this limit when one node loads, the loading task for all nodes is terminated. Before cluster loading and submission, check if the total number of errors exceeds the limit. If it exceeds the limit, abandon the submission and report an error to exit.
- DATE FORM: Used to specify the default format for the date column type, such as'% Y -% m -% d '.
- DATETIME Format: Used to specify the default format for the datetime column, such as'% Y -% m -% d% H:% i:% s'.
- TIMESTAMP Format: used to specify the default format for timestamp columns, such as'% Y -% m -% d% H:% i:% s.% f'。
- TIME FORM: Used to specify the default format for the time column, such as'% H:% i:% s'.
- TRACE: Used to indicate whether to save error data tracing during this load. If

specified as 0, no traceability is performed. If specified as 1, traceability is performed. The default value is 1.

Traceability information includes the file and line number where the error data is located.

- TRACE_ PATH: Used to specify the error data and log storage path generated during this loading process. This parameter only takes effect when the log aggregation function is disabled, with the default value being in the '$GBASE_BASE/log/gbase/loader_logs' section of the loading node.
- NOSPLIT: Used to specify whether to disable the block loading function in this loading task, and specifying this parameter will disable the block loading function. If this parameter is not specified, the block loading function will be automatically activated during cluster loading. The data will be evenly partitioned based on the amount of data and the number of loading nodes participating in the operation to balance the load on the data server and data processing nodes and optimize loading performance.
- PARALLEL: Used to control the parallelism of cluster loading, with a value range of [0104]. The default value is 0, which means that the parallelism value is the maximum number of available threads in the thread pool.
- MAX_ DATA_ PROCESS: Used to specify the maximum number of nodes involved in data parsing in this loading task, with a value range of [1, 4294967295] and a default value of 16.
- MIN_ CHUNK_ SIZE: Used to specify the minimum granularity of data partitioning in this loading task, with a value range of [1, 4294967295] and a default value of 64M.
- SKIP_ BAD_ FILE: Used to specify whether to continue loading data files that do not exist or do not have read permissions during this loading task. If specified as 0, the loading terminates with an error. If specified as 1, ignore the exception file and continue loading. The default value is 0.
- IGNORE NUM Lines: Configure this parameter. The loading tool will filter the headers of all specified data files during this load, skipping the first NUM lines (the number of rows occupied by the header) of each file. The range of NUM values is [0, MAX_UINT].
- FILE_ Format: Used to specify the format of the loaded file. Enumeration type parameter, with values of UNDEFINED, UNCOMPRESSED, GZIP, SNAPPY, LZO, and defaults to UNDEFINED. Specify as UNDEFINED, indicating that no format is specified and the file format is automatically determined based on the file suffix; Specify as UNCOMPRESSED, indicating that the file is loaded as plain text; Specify as GZIP, which means loading files in GZIP format; Specify as SNAPPY, which means loading files in SNAPPY format; Specify as LZO, which means loading the file in LZO format.

5.2.2.2 Use constraints

- When using fixed length loading mode, the value of FIELDS DEFINER must be specified.
- When using text loading mode, NULL_ The default value of Value is' N '.
- When using text loading, the row separator defaults to ' n'.
- When using text loading, if a column of data may contain row delimiters, you need to enter the 'HAVING LINES SEPARATOR' clause in SQL, and also enter 'ENCLOSED BY' to specify the field delimiter.
- When the username (user), password (password), host name (host), or file path (path) in the URL where the file list is loaded contain special characters listed in the following table, percentage encoding is required to replace the special characters.

  URL：scheme://[user:password@]host[:port]/path

  Percentage encoding=%+two character hexadecimal value of special characters

Table 5152 Explanation of Replacing Special Characters with Percentage Code

Special Characters	Percentage code	explain
%	%25	Require percentage encoding
:	%3A	Standard gen-delims require percentage encoding
/	%2F	
?	%3F	
#	%23	
[	%5B	
]	%5D	
@	%40	
!	%21	Standard sub delims suggest percentage encoding
$	%24	
&	%26	
'	%27	
(	%28	
)	%29	
*	%2A	
+	%2B	
,	%2C	
;	%3B	
=	%3D	
\	%5C	

"	%22	Suggested percentage encoding for characters not listed in the standard
<	%3C	
>	%3E	
Space	%20	

The following content is quoted from the standard RFC-3986. Although some reserved characters may not cause URI resolution problems, it is still recommended to use percent encoding for all reserved characters. For more detailed URI encoding rules, please refer to the standard RFC-3986 document.

a) Percent-Encoding

pct-encoded = "%" HEXDIG HEXDIG

b) Reserved Characters

reserved = gen-delims / sub-delims

gen-delims = ":" / "/" / "?" / "#" / "[" / "]" / "@"

sub-delims = "!" / "$" / "&" / "'" / "(" / ")"
/ "*" / "+" / "," / ";" / "="

c) Unreserved Characters

unreserved = ALPHA / DIGIT / "-" / "." / "_" / "~"

Example: FTP username is test, password is abc/def

Error:

```
gbase> load data infile ' ftp://test:abc/def @192.168.0.1/data/*.tbl' into table t data_ format 3;
```

correct:

```
gbase> load data infile ' ftp://test:abc%2Fdef @192.168.0.1/data/*.tbl' into table t data_ format 3;
```

- When the user name (user), password, host name or file path in the URL of the loading file list contain the special characters listed in the following table, the special characters need to be replaced by escape character.

Table -5153 Special characters are replaced by escape character

Special Characters	Escape character	explain
\	\\	Escape character required
'	\'	Escape character required

Note: If the special characters listed in the above table have been coded with percent sign, it is unnecessary to use escape character instead.

Example: FTP username is test, password is abc def

Error:

```
gbase> load data infile ' ftp://test:abc \ def@192.168.0.1 /data/*.tbl' into table t data_ format 3;
```

correct:

```
gbase> load data infile ' ftp://test:abc \\ def@192.168.0.1 /data/*.tbl' into table t data_ for
```

```
mat 3;
```

- There are inconsistencies between the loose mode processing rules and the text loading processing rules:

  1) When the specified line separator is n and auto is specified_ fill_ When using the column parameter, empty lines will also be treated as one line, filled with null, and will not be skipped;

  2) When there are empty values in the data, the input data is null and not the default value. Setting the default value has no effect on the loading result;

  3) Support automatic truncation of ultra wide columns;

  4) The delimiters and column separators in the data file are inconsistent with the settings. If the first column is of character type, the data will be truncated and stored, and the subsequent fields will be empty; If the first column is numerical, then all are incorrect data;

  5) Specify auto_ fill_ Column, automatically fills in when there are fewer columns. Regardless of whether the column definition has a default value, null values will be used to fill in the missing columns instead of the default value.

- In any loading mode, the length of the loading line exceeds gbase_ loader_ max_ line_ The maximum line length of 64M limited by the length (67108864) parameter will result in an error.

5.2.2.3 HDFS file loading instructions

- To support NameNode high availability when loading HDFS files, it is necessary to first set gbase before executing the load statement_ hdfs_ namenodes='acitve_ nn, standby_ Nn ', specifies the highly available NameNode host information for HDFS. (HDFS typically consists of two NameNodes and several DataNodes, with one NameNode in the Active state and the other in the Standby state.)

  Example:

  Before executing the load statement, set system parameters for HDFS that supports high availability.

```
gbase> set gbase_ hdfs_ namenodes="192.168.10.1,192.168.10.2";
```

  The user enters a loading statement, specifies the loading of an HDFS file, and specifies the correct HDFS NameNode host name (or IP address) and port number in the URL.

```
gbase> LOAD DATA INFILE ' hdp://hadoop @192.168.10.1:50070/data/test.tbl' INTO TA
BLE test.t;
```

  Note: As shown in the above example, currently compatible tools support escaping special characters in the/data/test.tbl section, and support file names containing the following special characters:

'(Space),'! ',' "', '#', '$', '%', '&', '"', '(', ')', '+', '-', '.', ':', ';', '<', '=', '>', '@', '[', '\', ']', '^', '_', '`', '{', '|', '}', '~'

The following special characters are not supported for escape:

Wildcard characters: '*', '?'

Path separator: '/'

The following special character combinations are not supported: '% (', ';' (spaces and semicolons), ' [', ']'

The special character '' must be written as' 'in the configuration file

- Parallel import and export of multiple Hadoop clusters:

  When multiple HDFS environments require parallel import and export from the cluster, set gbase_ hdfs_ The namenodes parameter is separated by '|' between NameNode groups of multiple sets of HDFS

  gbase_ hdfs_ namenodes='hdfs1_ acitve_ nn, hdfs1_ standby_ nn | hdfs2_ acitve_ nn, hdfs2_ standby_ nn '

```
gbase_ hdfs_ namenodes='192.168.1.1,192.168.1.2|192.168.2.1,192.168.2.2'
load data infile ' hdp://gbase @192.168.1.1/data/f.tbl' into table test.t
load data infile ' hdp://gbase @192.168.2.1/data/f.tbl' into table test.t
```

5.2.2.4 KAFKA Data Source Loading Instructions

- Using the Kafka cluster as the data source, each topic in the Kafka cluster corresponds to a table in 8a, and a standard URL is used to represent a topic data source in Kafka. The URL format is defined as follows:

  kafka://broker/topic [?duration=1000][#frombeginning]

  For example:

  kafka://192.168.146.20:9092/vct?duration=1000#frombeginning

  Broker: includes the IP and port number of a node in the kafka cluster. For example, 192.168.146.20:9092.

  Topic: The topic name of the data source corresponding to the table to be loaded, case sensitive. For example, VCT.

  duration：’?’ As a prefix, kafka, as a data stream, has no end of file, so it is divided into small tasks according to time. Each time a part of data is submitted, the duration parameter sets the time length of the loading task, in ms. Duration is not set or set to 0. When loading, kafka is treated as a file and read until the end of each partition in kafka, regardless of case.

  Frombedding: '#' is used as a prefix, and loading will automatically maintain the offset of each partition. The offset is monotonically increasing. If users need data to

be repeatedly stored, using 'from beginning' can reset the offset, which is not case sensitive. Offset is uniformly managed by loading. After each successful loading, the read location is persistence to the gclusterdb system library named topicname_ dbname_ In the table of tbname, the offset will be restored from the table the next time the load is started, and data will continue to be read from kafka.

Using Kafka clusters as data source loading can support the following functional features:

1. It can achieve streaming loading and load the data flowing into Kafka into 8a in real-time.

Streaming data loading requires users to repeatedly call loading statements using scripts to ensure that the data can be loaded into the database in a timely manner after entering Kafka. The frequency of loading submissions is guaranteed by the duration, and needs to be comprehensively considered based on the real-time and loading performance requirements of the business. It is recommended not to exceed 10 minutes.

while true

do

load data infile ' kafka://192.168.146.20:9092/vct?duration=2000#frombeginning '

into table vc1.testdb.t fields terminated by '|';

done

During the execution process, if the load fails, an error message will be displayed to the standard output, and the load will not stop. Users can determine whether to stop loading based on the usage scenario.

2. Supports high availability of kafka's broker

Add the parameter gcluster to the configuration file of Gcluster ($GCLUSTER_BASE/config/gbase_8a_gcluster. cnf)_ kafka_ Brokers, list multiple brokers: IPs of the Kafka cluster in commas. After saving the configuration file, restart the 8a cluster service to make the modifications effective. In this way, the broker specified by the URL in the loading statement can still be loaded offline and executed correctly. For example:

```
vi $GCLUSTER_ BASE/config/gbase_ 8a_ gcluster.cnf

gcluster_ kafka_ brokers =
'192.168.146.20:9092,192.168.146.21:9092,192.168.146.22:9092'

gcluster_ services all restart
```

Current gcluster_ kafka_ Brokers high availability only supports one set of Kafka clusters as the data source, and is not available when loading multiple sets of Kafka cluster data.

3. Support for Kafka cluster data source loading under Kerberos security authentication

Kerberos that supports Kafka uses the SASL/GSSAPI authentication mechanism to

authenticate with 8a cluster clients, enabling 8a to load Kafka cluster data sources under kerberos authentication.

This feature currently only supports a set of Kafka clusters with kerberos authentication function as the loading data source, and cannot be used simultaneously with HDFS kerberos authentication function.

5.2.2.5 ORC file loading instructions

- ORC file description

  The orc file is composed of strips. Each strip has a fixed size and is independent of each other. The strip contains three parts: index, data, and metadata. The data part is stored after encoding and compression.

  Minimum data storage unit for stripe orc files

  Stripefoot stores metadata for stripes

  Footer stores stripe information, data structure information, statistical information, etc. of ORC files

  Postscript stores the basic metadata information of ORC files

- Loading and storing ORC files

  The loading syntax remains the same as that of 8a, such as:

  load data infile ' http://gbase @192.168.6.6/orcfile/test.orc' into table orctest data_ format orc;

  Parameter support:

  1. Support data sources such as local, FTP, SFTP, HTTP, HDFS, and GBFS, similar to 8a regular loading

  2. Parameters that can be used normally: file_ list、character_ set、data_ format、null_ value、fields preserve blanks（fields preserve leading blanks、fields preserve trailing blanks）、autofill、table_ fields、max_ bad_ records、datetime format、date format、time format、timestamp format、trace、trace_ path、nosplit、max_ data_ processors、skip_ bad_ file、set

  3. The syntax can be passed and executed normally, but it does not actually work. Warnings parameters will be reported: having lines separator, fields terminated by, fields enclosed by, length, lines terminated by, min_ chunk_ size

  4. Unsupported parameters (error may be reported): ignore un m lines, file_ Format specifying gzip, snappy, or lzo will generate an error, while specifying uncompressed/undefined can load normally.

  5. If there is abnormal data in the ORC file, the loader_ Xxx under logs_ orc_ The metadata information of abnormal data (including file name, stripe index, row index, etc.) and abnormal data content will be recorded in the loader. log. Orc file loading

no longer records loader_ Error data file under logs, load_ The result file is updated normally, load_ The status is updated normally.

6.data_ Format is orc or data_ Format is 8

7. Loading orc files does not require file names and suffixes, but does not support gzip/snap/lzo compressed file loading for orc

8. ORC file loading currently does not support the composite data types of ORC, such as struct, union, list, map, and other basic data types.

9. ORC file loading supports block loading, which is enabled by default during loading. When specifying the nosplit parameter, the block function is not enabled. The partitioning of ORC files is based on the minimum unit of stripe, and is loaded in blocks based on stripe by default.

10. ORC file loading supports wildcard batch loading, that is, file_ The orc file name in the list can partially contain the wildcard character *. If there is a file error, the loading task will terminate.

5.2.2.6 Example of using cluster loading

5.2.2.6.1 Load FTP server files in text mode

Load the a.tbl file located on the FTP server as text, using default row and column separators.

Example

```
LOAD DATA INFILE ' ftp://127.0.0.1/data/a.tbl ' INTO TABLE test.t DATA_
FORMAT 3;
```

5.2.2.6.2 Load FTP server compressed files in text mode

Load the test. tbl. lzo compressed file located on the FTP server as text, using default row and column separators.

Example

```
LOAD DATA INFILE ' ftp://127.0.0.1/data/test.tbl.lzo ' INTO TABLE test.t
DATA_ FORMAT 3;
```

5.2.2.6.3 Specify a username and password to load the FTP server file

Load the a.tbl file located on the FTP server as text, using default row and column separators, and ftp://user:password @The host/path method specifies the username and password of the FTP server.

Example

```
LOAD DATA INFILE ' ftp://gbase:gbase @127.0.0.1/data/a.tbl' INTO TABLE test.t DATA_ FORMAT 3;
```

5.2.2.6.4 Load HTTP server compressed files in text mode

Load the b. tbl. gz compressed file located on the HTTP server as text, using default row and column separators.

Example

```
LOAD DATA INFILE ' http://127.0.0.1/data/b.tbl.gz ' INTO TABLE test.t DATA_ FORMAT 3;
```

5.2.2.6.5 Specify a username and password to load the HTTP server file

Function Description

Load the a.tbl file located on the HTTP server as text, using default row and column separators, and http://user:password @The host/path method specifies the username and password of the HTTP server.

Example

```
LOAD DATA INFILE ' http://gbase:gbase @127.0.0.1/data/a.tbl' INTO TABLE test.t DATA_ FORMAT 3;
```

5.2.2.6.6 Text loading of HDFS server compressed files

Load the a.tbl.snappy compressed file located on the HDFS server as text, using default row and column separators, and hdp://user @The host/path method specifies the username of the HDFS server.

Example

```
LOAD DATA INFILE ' hdp://gbase @127.0.0.1:50070/data/a.tbl.snappy' INTO TABLE test.t DATA_ FORMAT 3;
```

5.2.2.6.7 Load SFTP server files in text mode

Load the a.tbl file located on the SFTP server as text, using default row and column separators, and sftp://user:password @The host/path method specifies the username and password of the SFTP server.

Example

```
LOAD DATA INFILE ' sftp://gbase:gbase @127.0.0.1/data/a.tbl' INTO TABLE test.t DATA_ FORMAT 3;
```

5.2.2.6.8 Text loading of GBFS server files

Load the part.tbl file located on the GBFS server as text, using the default row separator and '|' column separator.

Example

```
gbase> load data infile ' gbfs://192.168.146.20//opt/ssbm/part.tbl ' into table part data_ format 3    FIELDS TERMINATED BY '|';
```

5.2.2.6.9 Load Kafka data source file

Use URL to load data from Kafka data source, using default row delimiter and '|' column delimiter.

Example

```
gbase> load data infile '
kafka://192.168.146.20:9092/vct?duration=1000#frombeginning ' into table
vc1.testdb.t fields terminated by '|';
Query OK, 4 rows affected
Task 3089 finished, Loaded 4 records, Skipped 0 records
gbase> load data infile ' kafka://192.168.146.20:9092/vct?duration=1000 ' into
table vc1.testdb.t fields terminated by '|';
Query OK, 1 row affected
Task 3090 finished, Loaded 1 records, Skipped 0 records
gbase> select * from gclusterdb.vct_ testdb_ t;
+------+------------------+---------------------+
| scn    | partition_ offset | commit_ time                |
+------+------------------+---------------------+
| 3088 | 0:-2                       | 2021-07-01 17:31:01 |
| 3089 | 0:4                        | 2021-07-01 17:42:31 |
| 3090 | 0:5                        | 2021-07-01 17:45:50 |
+------+------------------+---------------------+
7 rows in set (Elapsed: 00:00:01.02)
```

5.2.2.6.10 Multiple data file loading

Specify multiple data source files, which may not be on the same file server, separated by commas.

Example

```
gbase> LOAD DATA INFILE ' ftp://192.168.0.1/pub/lineitem.tbl ,
http://192.168.0.2/lineitem.tbl ' INTO TABLE test.lineitem FIELDS
TERMINATED BY '|' ENCLOSED BY '"' LINES TERMINATED    BY '\n';
Query OK, 24000000 rows affected
Task 1 finished, Loaded 24000000 records, Skipped 0 records
```

5.2.2.6.11 Loading multiple data files using wildcards

By using wildcards in the loading file name, multiple files in the same directory can be specified. Multiple file names containing wildcards can be specified in a single load statement. Currently, it supports the use of *, *. tbl style wildcards in file names, which can be used in the FTP/HTTP/HDFS/SFTP protocol. Wildcards can also be used in the loading path, and currently support the use of * The wildcard character of the [] style requires the user to have executable permissions on the directory when wildcarding the path.

FTP servers that support wildcard loading require support for the LIST command. Both vsftpd and IIS FTP have been tested to meet this requirement.

SFTP servers that support wildcard loading have been tested for sshd to meet this requirement.

HTTP servers that support wildcard loading require them to respond to directory requests. Due to the different responses of different HTTP servers to directory requests, some HTTP servers may not support loading data files using wildcards. Both Apache HTTP and IIS HTTP have been tested to meet this requirement.

Example

```
gbase> LOAD DATA INFILE ' ftp://192.168.10.114/data/ *' INTO TABLE
test.t;
Query OK, 100000 rows affected
Task 1 finished, Loaded 100000 records, Skipped 0 records
gbase> LOAD DATA INFILE ' sftp://gbase:gbase @192.168.10.114/data/*'
INTO TABLE test.t;
Query OK, 100000 rows affected
Task 1 finished, Loaded 100000 records, Skipped 0 records
gbase> LOAD DATA INFILE ' http://192.168.10.114/data/ *' INTO TABLE
test.t;
Query OK, 100000 rows affected
```

```
Task 1 finished, Loaded 100000 records, Skipped 0 records
gbase> LOAD DATA INFILE ' hdp://gbase @127.0.0.1:50070/data/*' INTO TABLE test.t;
Query OK, 100000 rows affected
Task 1 finished, Loaded 100000 records, Skipped 0 records
```

5.2.2.6.12 Load with specified delimiters, column separators, and row separators

Example

```
Partial data files:
1|1551894|76910|1|17|33078.94|0.04|0.02|N|O|1996-03-13|1996-02-12|1996-03-22|
"DELIVER IN PERSON"|TRUCK|egular courts above the|
1|673091|73092|2|36|38306.16|0.09|0.06|N|O|1996-04-12|1996-02-28|1996-04-20|"
TAKE BACK RETURN"|MAIL|ly final dependencies: slyly bold |
Loading process:
gbase> LOAD DATA INFILE ' ftp://192.168.0.1/pub/lineitem.tbl ' INTO TABLE test.lineitem FIELDS TERMINATED BY '|' ENCLOSED BY '"' LINES TERMINATED BY '\n';
Query OK, 12000000 rows affected
Task 1 finished, Loaded 12000000 records, Skipped 0 records
```

5.2.2.6.13 Specify the format of the TIMESTAMP column in the data file

Example

```
Table creation statement:
CREATE TABLE ttimestamp(a TIMESTAMP DEFAULT '2014-01-01 12:25:36' ,b INT);
Data file:
2014-01-01 12:01:01|1
|2
|3
|4
|5
2014-01-02 12:03:03|6
Loading process:
gbase> LOAD DATA INFILE ' http://10.10.120.226/timestamp.txt ' INTO TABLE test.ttimestamp DATA_ FORMAT 3 FIELDS TERMINATED BY '|' TIMESTAMP FORMAT '%Y-%m-%d %H:%i:%s';
Query OK, 6 rows affected
Task 2 finished, Loaded 6 records, Skipped 0 records
```

Query inbound data:

gbase> SELECT * FROM ttimestamp ORDER BY b

```
+---------------------+------+
| a                   | b    |
+---------------------+------+
| 2014-01-01 12:01:01 |    1 |
| 2014-01-01 12:25:36 |    2 |
| 2014-01-01 12:25:36 |    3 |
| 2014-01-01 12:25:36 |    4 |
| 2014-01-01 12:25:36 |    5 |
| 2014-01-02 12:03:03 |    6 |
+---------------------+------+
```

5.2.2.6.14 Using TABLE_ FIELDS specifies the loading column and date format

Example

Table creation statement:

CREATE TABLE t (i INT, vc VARCHAR(10), dt DATETIME DEFAULT '2000-01-01 00:00:01', dt1 DATETIME DEFAULT '2000-01-01 00:00:01');

Data file:

31589,E,02:02:02 2094-12-13,2082-12-24 01:01:01
16993,jcWaz,02:02:02 2060-10-22,2037-11-17 01:01:01
7584,jubNKAmT,02:02:02 2058-12-24,2066-11-26 01:01:01
8698,iOStkY,02:02:02 2024-11-17,2064-10-25 01:01:01
23256,itWsHqL,02:02:02 2069-10-24,2021-11-19 01:01:01
21932,GelDJbuE,02:02:02 2017-11-26,2075-11-19 01:01:01
4859,Gl,02:02:02 2040-10-16,2051-10-25 01:01:01
11751,InTUcdIM,02:02:02 2048-12-23,2099-10-26 01:01:01
8487,JZ,02:02:02 2026-12-13,2084-11-15 01:01:01
3693,lEKyI,02:02:02 2063-10-21,2026-11-20 01:01:01

Loading process:

gbase> LOAD DATA INFILE ' ftp://192.168.88.141/load_data/table_fields.tbl ' INTO TABLE test.t fields terminated by ',' TABLE_ FIELDS 'i, vc, dt date "%H:%i:%s %Y-%m-%d", dt1 date "%Y-%m-%d %H:%i:%s"';

Query OK, 10 rows affected, 0 warnings (Elapsed: 00:00:01.48)

Task 1114 finished, Loaded 10 records, Skipped 0 records

Query inbound data:

gbase> SELECT * FROM Test.t;

```
+-------+----------+---------------------+---------------------+
| i     | vc       | dt                  | dt1                 |
+-------+----------+---------------------+---------------------+
```

```
| 23256 | itWsHqL    | 2069-10-24 02:02:02 | 2021-11-19 01:01:01 |
| 31589 | E          | 2094-12-13 02:02:02 | 2082-12-24 01:01:01 |
|  8487 | JZ         | 2026-12-13 02:02:02 | 2084-11-15 01:01:01 |
| 21932 | GelDJbuE | 2017-11-26 02:02:02 | 2075-11-19 01:01:01 |
| 16993 | jcWaz      | 2060-10-22 02:02:02 | 2037-11-17 01:01:01 |
|  3693 | lEKyI      | 2063-10-21 02:02:02 | 2026-11-20 01:01:01 |
| 11751 | InTUcdIM | 2048-12-23 02:02:02 | 2099-10-26 01:01:01 |
|  8698 | iOStkY     | 2024-11-17 02:02:02 | 2064-10-25 01:01:01 |
|  4859 | Gl         | 2040-10-16 02:02:02 | 2051-10-25 01:01:01 |
|  7584 | jubNKAmT | 2058-12-24 02:02:02 | 2066-11-26 01:01:01 |
+-------+----------+---------------------+---------------------+
10 rows in set
```

5.2.2.6.15 Using TABLE_ FIELDS specifies ignored fields in the loading data file

- The field data of the data file corresponding to the columns defined in the table needs to be ignored, not loaded into the database, and all fields should be filled with NULL, using the filler keyword.

```
gbase@suse100-4 :~> cat t1.txt
1|a|b

gbase> create table t1(c1 int,c2 varchar(10),c3 varchar(20));
Query OK, 0 rows affected (Elapsed: 00:00:00.06)

Add filler to existing column names
gbase> load data infile ' sftp://gbase:gbase
@192.168.105.54//home/gbase/t1.txt' into table t1 fields terminated by '|'
table_ fields 'c1,c2 filler,c3';
Query OK, 1 row affected (Elapsed: 00:00:01.13)
Task 1310 finished, Loaded 1 records, Skipped 0 records

gbase> select * from t1;
+------+------+------+
| c1   | c2   | c3   |
+------+------+------+
|    1 | NULL | b    |
+------+------+------+
3 rows in set (Elapsed: 00:00:00.03)
```

- The data file has more fields than the table defined columns, and any additional columns need to be ignored

```
gbase@suse100-4 :~> cat t1.txt
1|m|a|b
```

```
gbase> create table t1(c1 int,c2 varchar(10),c3 varchar(20));
Query OK, 0 rows affected (Elapsed: 00:00:00.06)

Add filler to existing column names
gbase> load data infile ' sftp://gbase:gbase @192.168.105.54//home/gbase/t1.txt'
into table t1 fields terminated by '|' table_ fields 'c1,m,c2,c3';
Query OK, 1 row affected (Elapsed: 00:00:01.13)
Task 1310 finished, Loaded 1 records, Skipped 0 records

gbase> select * from t1;
+------+------+------+
| c1   | c2   | c3   |
+------+------+------+
|    1 | a    | b    |
+------+------+------+
3 rows in set (Elapsed: 00:00:00.03)
```

5.2.2.6.16 Using TABLE_ FIELDS specifies loading longblob data

There are two methods for loading longblob type data: one is in the form of data, and the other is in the form of files. When loading this type of data, it is necessary to enter the table_ Specify type for the longblob column in the fields parameter_ text、type_ base64、type_ The URL parameter is specified as follows:

- Data format, the data file that needs to be loaded contains longblob type data.

```
gbase>load data infile ‘ http://192.168.6.39/test.tbl ’ into table data_ test fields
terminated by ‘|’ table_ fields ‘a,b,c type_ text,d’;
```

In SQL, test.tbl is the file that needs to be loaded, and it contains longblob data.

1) The data has not undergone any encoding and needs to be processed through a table in SQL_ Fields specifies the type for the longblob column_ Text parameter. For example:

Table creation statement:

CREATE TABLE data_ test (column_1 INT, column_2 VARCHAR(10), column_ 3 LONGBLOB, column_ 4 VARCHAR(10));

Data file:

123|eqwerqwee|asdfsacq|adfasdfaa
234|qreqwerqw|sfwrwers|asfdasdfa

Loading process:

gbase>LOAD DATA INFILE ‘ http://192.168.6.39/test.tbl ’ INTO TABLE data_ test FIELDS TERMINATED BY ‘|’ TABLE_ FIELDS ‘column_ 1, column_ 2, column_ 3 type_ text, column_ 4’;

```
Query OK, 2 rows affected (Elapsed: 00:00:00.11)
Task 13 finished, Loaded 2 records, Skipped 0 records
Query inbound data:
gbase> SELECT * FROM test.data_ test;
+------+-----------+----------+-----------+
|column_1 | column_2 | column_3 | column_4 |
+------+-----------+----------+-----------+
|   123 | eqwerqwee | asdfsacq | adfasdfaa |
|   234 | qreqwerqw | sfwrwers | asfdasdfa |
2 rows in set
```

2) Longblob data is encoded in base64 format and needs to be processed through table in SQL_ Fields specifies the type for the longblob column_ Base64 parameter. For example:

```
Table creation statement:
CREATE TABLE data_ test (column_1 INT, column_2 VARCHAR(10),
column_ 3 LONGBLOB, column_ 4 VARCHAR(10));
Data file:
123|eqwerqwee| PQEWIIAZX==|adfasdfaa
234|qreqwerqw| PQEWIIAZX==|asfdasdfa
Loading process:
gbase> LOAD DATA INFILE ' http://192.168.6.39/test.tbl ' INTO TABLE
data_ test FIELDS TERMINATED BY '|' TABLE_ FIELDS   'column_ 1,
column_ 2, column_ 3 type_ base64, column_ 4';
Query OK, 2 rows affected (Elapsed: 00:00:00.11)
Task 14 finished, Loaded 2 records, Skipped 0 records
Query inbound data:
gbase> SELECT * FROM test.data_ test;
    +----------+-----------+----------+-----------+
    | column_ 1 | column_ 2   | column_ 3 | column_ 4   |
    +----------+-----------+----------+-----------+
    |          234 | qreqwerqw | = ®       | asfdasdfa |
    |          123 | eqwerqwee | = ®       | adfasdfaa |
    +----------+-----------+----------+-----------+
2 rows in set
```

- File format

  The loading method for longblob files is as follows:

```
gbase>Load data infile ‘ http://192.168.6.39/test.tbl ’ into table data_ test fields
terminated by ‘|’ table_ fields ‘a,b,c type_ url,d’;
```

The longblob column in test.tbl writes the path to the longblob file to be loaded, which can be an absolute path (such as http/ftp/sftp/hdp protocol type) or a relative path.

In SQL, test.tbl is the file that needs to be loaded. In the following example, test_

url.jpg、test_ url_ 1.jpg、test_ url_ 2. jpg is a longblob file, pointing to test_ The path of url.jpg is an absolute path; Point to test_ url_ 1.jpg、test_ url_ 2. The relative path of jpg. test_ url_ 1. jpg and test.tbl are in the same directory, test_ url_ 2. jpg is located in the test folder (which is in the same directory as test.tbl). When loading the above jpg file, it is necessary to write the path to the file in the longblob column of test.tbl, such as:

```
123|eqwerqwee| http://192.168.6.11/test_url.jpg          |adfasdfaa
234|qreqwerqw|test_ url_ 1.jpg                            |asfdasdfa
234|qreqwerqw|./ test_ url_ 1.jpg                         |asfdasdfa
123|qwerwesqw|test/test_ url_ 2.jpg                       |xcvb
```

5.2.2.6.17 Using the AUTOFILL keyword to fill in missing data

Example

Table creation statement:

```
CREATE TABLE t(a int,b VARCHAR(10),c VARCHAR(10));
```

Data file:

```
1|first
2|second
```

Loading process:

```
gbase> LOAD DATA INFILE ' ftp://192.168.88.141/load_data/autofill.tbl '
INTO TABLE test.t FIELDS TERMINATED BY '|'    AUTOFILL;
Query OK, 2 rows affected, 3 warnings (Elapsed: 00:00:00.84)
Task 1107 finished, Loaded 2 records, Skipped 0 records
```

Query inbound data:

```
gbase> SELECT * FROM dual;
+------+--------+------+
| a       | b           | c       |
+------+--------+------+
|       2 | second | NULL |
|       1 | first    | NULL |
+------+--------+------+
2 rows in set
```

5.2.2.6.18 Fixed length text loading

Example

Table creation statement:

```
CREATE TABLE t (i INT, vc VARCHAR(10), c CHAR(10));
```

Data file:

```
09912141332156508668763659755 3
58723137484370618221181662539 1
37190916968790186599843824854 0
76808451762094544568037503391 8
52101996126985548145450309367 9
87209812028354897319045976713 9
04745877273277605373081527324 8
74919598277905983239136757045 4
77378114077358138024016623816 7
73308614740291835211377307570 9
```

Loading process:

```
gbase> LOAD DATA INFILE ' ftp://192.168.88.141/load_data/definer.tbl '
INTO TABLE test.t DATA_ FORMAT 4 FIELDS LENGTH'1,10,10';
Query OK, 10 rows affected
Task 1128 finished, Loaded 10 records, Skipped 0 records
```

Query inbound data:

```
gbase> select * FROM dual;
+------+------------+------------+
| i    | vc         | c          |
+------+------------+------------+
|    3 | 7190916968 | 7901865998 |
|    0 | 4745877273 | 2776053730 |
|    0 | 9912141332 | 1565086687 |
|    7 | 7378114077 | 3581380240 |
|    5 | 8723137484 | 3706182211 |
|    7 | 3308614740 | 2918352113 |
|    7 | 6808451762 | 0945445680 |
|    5 | 2101996126 | 9855481454 |
|    8 | 7209812028 | 3548973190 |
|    7 | 4919598277 | 9059832391 |
+------+------------+------------+
10 rows in set
```

5.2.2.6.19 Loading data files using IGNORE NUM LINES

Example

Table creation statement:

```
CREATE TABLE data_ test (column_1 INT, column_2 VARCHAR(50),
column_ 3 VARCHAR(20), column_ 4 VARCHAR(10));
```

Data file:

```
123|eqwerqwee|asdfsacq|adfasdfaa
234|qreqwerqw|sfwrwers|asfdasdfa
435|asdfsfadfasf|werqqws|asdfasfds
765|ertyertyeref|fdwaesws|sfgwerwr
Loading process:
gbase>LOAD DATA INFILE ‘ http://192.168.6.39/test.tbl ’ INTO TABLE
data_ test FIELDS TERMINATED BY ‘|’ IGNORE 3 LINES;
Task 26 finished, Loaded 1 records, Skipped 0 records
Query inbound data:
gbase> SELECT * FROM test.data_ test;
+------+----------+----------+----------+
|column_1 | column_2 | column_3 | column_4 |
+------+----------+----------+----------+
|    765 | ertyertyeref | fdwaesws | sfgwerwr |
1 row in set
```

5.2.2.6.20 Loading data files using SET

Example

```
Table creation statement:
CREATE TABLE "t" ("a" varchar(10) DEFAULT NULL, "b" int(11)
DEFAULT NULL, "c" datetime DEFAULT NULL, "d" varchar(10)
DEFAULT NULL, "e" decimal(10,2) DEFAULT NULL );
Data file:
Hello|01
Good|02
Better|03
Loading process:
gbase> LOAD DATA INFILE ' http://192.168.6.39/data.tbl ' INTO TABLE t
FIELDS TERMINATED BY '|' SET c='2016-06-06
18:08:08',d='default',e=20.6;
Query OK, 3 rows affected
Task 2920 finished, Loaded 3 records, Skipped 0 records
Query inbound data:
gbase> SELECT * FROM t;
+--------+------+---------------------------------+---------+-------+
| a           | b       | c                                | d             | e         |
+--------+------+---------------------------------+---------+-------+
| Hello    |        1 | 2016-06-06 18:08:08 | default | 20.60 |
| Good     |        2 | 2016-06-06 18:08:08 | default | 20.60 |
| Better |        3 | 2016-06-06 18:08:08 | default | 20.60 |
+--------+------+---------------------------------+---------+-------+
```

```
5rows in set
```

5.2.2.6.21 Use NULL_ Values specifies the null character to load the data file

Example

Table creation statement:

```
CREATE TABLE "test" (    "column_1" int(11) DEFAULT NULL,
"column_2" varchar(10) DEFAULT NULL,    "column_3" varchar(20)
DEFAULT NULL);
```

Data file:

```
43452|sisoekso|mozoa,a
59432|gg|laqpqpd
03890|lqps,rpd|gg
```

Loading process:

```
gbase> LOAD DATA INFILE ' http://192.168.153.32/1.txt ' INTO TABLE
test FIELDS TERMINATED BY '|' NULL_ VALUE 'gg';
Query OK, 3 rows affected
Task 25 finished, Loaded 3 records, Skipped 0 records
```

Query inbound data:

```
gbase> SELECT * FROM test;
+----------+----------+----------+
| column_ 1 | column_ 2 | column_ 3 |
+----------+----------+----------+
|       59432 | NULL    | laqpqpd    |
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | NULL         |
+----------+----------+-----------+
3 rows in set
```

5.2.2.6.22 Using PRESERVE BLANKS to specify null characters to load data files

Example

Table creation statement:

```
CREATE TABLE "test_1" (    "column_1" int(11) DEFAULT NULL,
"column_2" varchar(10) DEFAULT NULL,    "column_3" varchar(20)
DEFAULT NULL);
```

Data file:

```
43452 | sisoekso | mozoa,a
59432 | gg|laqpqpd
03890 | lqps,rpd|gg
```

```
Loading process:
gbase> LOAD DATA INFILE ' http://192.168.153.32/1.txt ' INTO TABLE test_ 1 FIELDS TERMINATED BY '|' PRESERVE BLANKS;
Query OK, 3 rows affected
Task 27 finished, Loaded 3 records, Skipped 0 records
Query inbound data:
gbase> SELECT * FROM test_ 1;
+----------+------------+-----------+
| column_ 1 | column_ 2     | column_ 3    |
+----------+------------+-----------+
|       43452 |    sisoekso    |    mozoa,a    |
|        3890 |    lqps,rpd    | gg              |
|       59432 |    gg              | laqpqpd     |
+----------+------------+-----------+
3 rows in set
```

5.2.2.6.23 Using MAX_ BAD_ RECORDS Load Data File

Example

```
Table creation statement:
CREATE TABLE "test_2" (    "column_1" int(11) DEFAULT NULL, "column_2" varchar(10) DEFAULT NULL,    "column_3" varchar(20) DEFAULT NULL);
Data file:
43452|sisoekso|mozoa,a
59432|gg|laqpqpd
03890|lqps,rpd|gg
Loading process:
gbase> LOAD DATA INFILE ' http://192.168.153.32/1.txt ' INTO TABLE test_ 2 FIELDS TERMINATED BY '|' MAX_ BAD_ RECORDS 1;
ERROR 1733 (HY000): (GBA-01EX-700) Gbase general error: Task 268 failed, [192.168.153.32:5050](GBA-02AD-0005)Failed to query in gnode:
DETAIL: (GBA-01-600) Gbase internal error: Task 268, Too many bad records!
SQL: LOAD /*+ TID('11471') */ DATA INFILE ' http://192.168.153.32/1.txt#offset=0&length=58&firstblock&ffsize=58 ' INTO TABLE `test`.` test_ 2_ n1` DATA_ FORMAT 3 FIELDS TERMINATED BY ',' MAX_ BAD_ RECORDS 1 HOST '192.168.153.32' CURRENT_ TIMESTAMP 1510598427 SCN_ NUMBER 268 GCLUSTER_ PORT 5258 INTO SERVER
Query inbound data:
gbase> SELECT * FROM test_ 2;
Empty set (Elapsed: 00:00:00.03)
```

5.2.2.6.24 Using SKIP_ BAD_ FILE Load Data File

Example

Table creation statement:

```
CREATE TABLE "test_3" (    "column_1" int(11) DEFAULT NULL,
"column_2" varchar(10) DEFAULT NULL,    "column_3" varchar(20)
DEFAULT NULL);
```

Data file, 1. txt content is completely consistent with 2. txt:

```
-rw-r--r-- 1 root root      58 Nov 13 09:13 1.txt
--w------- 1 root root      58 Nov 13 09:21 2.txt
```

Specify skip_ bad_ File is 0:

```
gbase>LOAD DATA INFILE ' http://192.168.153.32/ *.txt' INTO TABLE
test_ 3 FIELDS TERMINATED BY '|' SKIP_ BAD_ FILE 0;
ERROR 1733 (HY000): (GBA-01EX-700) Gbase general error: Expanding
wildcard operation failed with error - I/O operation on http://192.168.153.32/2.txt
failed with error - Access denied to remote resource, HTTP/1.1 403 Forbidden,
File name http://192.168.153.32/2.txt    uri : http://192.168.153.32/%2a.txt.
```

Query inbound data:

```
gbase> SELECT * FROM test_ 3;
Empty set (Elapsed: 00:00:00.00)
```

Specify skip_ bad_ File is 1:

```
gbase> LOAD DATA INFILE ' http://192.168.153.32/ *.txt' INTO TABLE
test_ 3 FIELDS TERMINATED BY '|' SKIP_ BAD_ FILE 1;
Query OK, 3 rows affected (Elapsed: 00:00:00.58)
Task 42 finished, Loaded 3 records, Skipped 0 records, Ignored 1 files
```

Query inbound data:

```
gbase> SELECT * FROM test_ 3;
+----------+----------+----------+
| column_ 1 | column_ 2 | column_ 3 |
+----------+----------+----------+
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | gg             |
|       59432 | gg             | laqpqpd    |
+----------+----------+----------+
3 rows in set (Elapsed: 00:00:00.01)
```

5.2.2.6.25 Using MIN_ CHUNK_ SIZE Load Data File

Example

Table creation statement:

```
CREATE TABLE "test_4" (    "column_1" int(11) DEFAULT NULL,
"column_2" varchar(10) DEFAULT NULL,    "column_3" varchar(20)
DEFAULT NULL);
Data file:
43452|sisoekso|mozoa,a
59432|gg|laqpqpd
03890|lqps,rpd|gg
Loading process:
gbase> LOAD DATA INFILE ' http://192.168.153.32/1.txt ' INTO TABLE
test_ 4 FIELDS TERMINATED BY '|' MIN_ CHUNK_ SIZE 33554432;
Query OK, 3 rows affected (Elapsed: 00:00:00.31)
Task 252 finished, Loaded 3 records, Skipped 0 records
Query inbound data:
gbase> SELECT * FROM test_ 4;
+----------+----------+----------+
| column_ 1 | column_ 2 | column_ 3 |
+----------+----------+----------+
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | gg             |
|       59432 | gg             | laqpqpd    |
+----------+----------+----------+
3 rows in set (Elapsed: 00:00:00.01)
```

5.2.2.6.26 Using MAX_ DATA_ PROCESS Loading Data Files

Example

```
Table creation statement:
CREATE TABLE "test_5" (    "column_1" int(11) DEFAULT NULL,
"column_2" varchar(10) DEFAULT NULL,    "column_3" varchar(20)
DEFAULT NULL);
Data file:
43452|sisoekso|mozoa,a
59432|gg|laqpqpd
03890|lqps,rpd|gg
Loading process:
gbase> LOAD DATA INFILE ' http://192.168.153.32/1.txt ' INTO TABLE
test_ 5 FIELDS TERMINATED BY '|' MAX_ DATA_ PROCESSORS 2;
Query OK, 3 rows affected (Elapsed: 00:00:00.52)
Task 253 finished, Loaded 3 records, Skipped 0 records
Query inbound data:
gbase> SELECT * FROM test_ 5;
```

```
+----------+----------+----------+
| column_ 1 | column_ 2 | column_ 3 |
+----------+----------+----------+
|       43452 | sisoekso | mozoa,a    |
|        3890 | lqps,rpd | gg             |
|       59432 | gg             | laqpqpd    |
+----------+----------+----------+
3 rows in set (Elapsed: 00:00:00.01)
```

5.2.2.6.27Loading data files using PARALLEL

Example

Table creation statement:

```
CREATE TABLE "test_6" (    "column_1" INT(11) DEFAULT NULL,
"column_2" VARCHAR(10) DEFAULT NULL,    "column_3" varchar(20)
DEFAULT NULL);
```

Data file:

```
43452|sisoekso|mozoa,a
59432|gg|laqpqpd
03890|lqps,rpd|gg
```

Loading process:

```
gbase> LOAD DATA INFILE ' http://192.168.153.32/1.txt ' iNTO TABLE
test_ 6 FIELDS TERMINATED BY '|' PARALLEL 2;
```

Query inbound data:

```
gbase> SELECT * FROM test_ 6;
+----------+----------+----------+
| column_ 1 | column_ 2 | column_ 3 |
+----------+----------+----------+
|       43452 | sisoekso | mozoa,a    |
|        3890 | lqps,rpd | gg             |
|       59432 | gg             | laqpqpd    |
+----------+----------+----------+
3 rows in set (Elapsed: 00:00:00.01)
```

5.2.2.6.28Loading data files using NOSPLIT

Example

Table creation statement:

```
CREATE TABLE "test_7" (    "column_1" int(11) DEFAULT NULL,
"column_2" varchar(10) DEFAULT NULL,    "column_3" varchar(20)
```

```
DEFAULT NULL);
Data file:
43452|sisoekso|mozoa,a
59432|gg|laqpqpd
03890|lqps,rpd|gg
Loading process:
gbase> LOAD DATA INFILE ' http://192.168.153.32/1.txt ' INTO TABLE test_ 7 FIELDS TERMINATED BY '|'  NOSPLIT;
Query OK, 3 rows affected (Elapsed: 00:00:00.16)
Task 256 finished, Loaded 3 records, Skipped 0 records
Query inbound data:
gbase> SELECT * FROM test_ 7;
+----------+----------+----------+
| column_ 1 | column_ 2 | column_ 3 |
+----------+----------+----------+
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | gg             |
|       59432 | gg             | laqpqpd    |
+----------+----------+----------+
3 rows in set (Elapsed: 00:00:00.01)
```

5.2.2.6.29 Loading data files using CHARACTER SET

Example

```
Table creation statement:
CREATE TABLE "test_8" (  "column_1" INT(11) DEFAULT NULL, "column_2" VARCHAR(10) DEFAULT NULL,  "column_3" VARCHAR(20) DEFAULT NULL);
Data file:
43452|sisoekso|mozoa,a
59432|gg|laqpqpd
03890|lqps,rpd|gg
Loading process:
gbase> LOAD DATA INFILE ' http://192.168.153.32/1.txt ' INTO TABLE test_ 8  CHARACTER SET gbk FIELDS TERMINATED BY '|';
Query OK, 3 rows affected (Elapsed: 00:00:00.19)
Task 258 finished, Loaded 3 records, Skipped 0 records
Query inbound data:
gbase> SELECT * FROM test_ 8;
+----------+----------+----------+
| column_ 1 | column_ 2 | column_ 3 |
```

```
+----------+----------+----------+
|       43452 | sisoekso | mozoa,a    |
|        3890 | lqps,rpd | gg             |
|       59432 | gg             | laqpqpd    |
+----------+----------+----------+
3 rows in set (Elapsed: 00:00:00.01)
```

5.2.2.6.30 Using TRACE_ PATH Load Data File

Example

Table creation statement:

CREATE TABLE "test_9" ("column_1" int(11) DEFAULT NULL, "column_2" varchar(10) DEFAULT NULL, "column_3" varchar(20) DEFAULT NULL);

Data file:

43452|sisoekso|mozoa,a
59432|gg|laqpqpd
03890|lqps,rpd|gg

Loading process:

gbase> LOAD DATA INFILE ' http://192.168.153.32/1.txt ' INTO TABLE test_ 9 FIELDS TERMINATED BY '||' TRACE 1 TRACE_ PATH '/home/gbase/test/';
Query OK, 0 rows affected (Elapsed: 00:00:00.19)
Task 264 finished, Loaded 0 records, Skipped 3 records

Query log file:

[gbase@localhost 11]$ pwd
/home/gbase/test
[gbase@localhost 11]$ ll
total 8
-rw-rw---- 1 gbase gbase 58 Nov 13 13:36 267_ test_ test_ 2_ n1_ 192.168.153.32_ 20171113133614.err
-rw-rw---- 1 gbase gbase 392 Nov 13 13:36 267_ test_ test_ 2_ n1_ 192.168.153.32_ 20171113133614.trc

5.2.2.6.31 Load S3 server files in text mode

Load the a.tbl file located on the S3 server as text, using default row and column separators.

Example

```
LOAD DATA INFILE
```

```
's3n://GPCQN6HKP2BI3N6NKZGY:Nkf5ad6WD2MbWF6F6GD
obB8NudwC58ist% 2FJNJwY0@127.0.0.1 :9050/us-east-1/loaddata/a.tbl;
```

5.2.2.6.32 Loading S3 server files using wildcards

Load a set of files located on the S3 server in text format, using default row and column separators, and wildcard loading.

Example

```
LOAD DATA INFILE
's3n://GPCQN6HKP2BI3N6NKZGY:Nkf5ad6WD2MbWF6F6GDobB8N
udwC58ist% 2FJNJwY0@127.0.0.1 :9050/us-east-1/loaddata/test/*.tbl' INTO
TABLE test.t;
```

5.2.2.6.33 Local file loading on several specified nodes

use file://host +abs_ Path specifies file_ List information, multiple file://host +abs_ Separate paths with commas, abs_ Path contains wildcard characters. The definition format of a URL is as follows:

```
file://host    +abs_ path[, file://host    +abs_ path]
```

Example

```
LOAD DATA INFILE ‘ file://192.168.6.72/var/ftp/pub/line5 *.tbl,
file://192.168.6.73/home/gbase/lineitem. *’ INTO TABLE test.t FIELDS
TERMINATED BY ‘|’;
```

5.2.2.6.34 Local file loading on all nodes of the cluster

Using file://+abs_ Path specifies file_ List information. Multiple files://+abs_ Use commas to separate paths. The definition format of a URL is as follows:

```
file:// +abs_ path[,file:// +abs_path]
```

Example

```
LOAD DATA INFILE ‘ file:///var/ftp/pub/line5 *.tbl, file:///home/gbase/lineitem.
*’ INTO TABLE test.t FIELDS TERMINATED BY ‘|’;
```

5.2.2.6.35 Use wildcards to load local files

For abs in URLs_ Path, supports wildcard function.

The definition format of a URL is as follows:

```
file://host    +abs_ path[, file://host    +abs_ Path] contains wildcard characters
```

Example

```
LOAD DATA INFILE ‘ file://192.168.6.72/var/ftp/pub/line5 *.tbl,
file://192.168.6.72/home/gbase/lineitem. *’ INTO TABLE test.t FIELDS
TERMINATED BY ‘|’;
```

5.2.2.6.36 Using FILE_ FORMAT loading data file

Using FILE_ Format parameter loading data, file_ All files in the list are processed in the same way.

Load Example

- Load GZIP format files, all files are processed according to this type, and the loading methods for SNAPPY and LZO files are similar.

```
LOAD DATA INFILE ‘ http://192.168.6.39/test.tbl.gz ,
http://192.168.6.39/test.tbl.gz001 ’ INTO TABLE test.t FIELDS TERMINATED
BY ‘|’ file_ format gzip;
```

- Load non compressed plain text, and all files are processed according to this type.

```
LOAD DATA INFILE ‘ http://192.168.6.39/test.tbl ,
http://192.168.6.39/test.tbl.lzo ’ INTO TABLE test.t FIELDS TERMINATED BY
‘|’ FILE_ FORMAT uncompressed;
```

- Parse the file format according to the file name suffix, and all files are processed according to this type.

```
LOAD DATA INFILE ‘ http://192.168.6.39/test.snappy ,
http://192.168.6.39/test.tbl ’ INTO TABLE test.t FIELDS TERMINATED BY ‘|’
FILE_ FORMAT undefined;
```

5.2.2.7 Cluster loading supports multi column hash tables

The load data syntax on the cluster supports multi column hashing, and the syntax remains unchanged.

Example

Table creation statement:

```
create table x0( entry_id int, id2 int, id3 int,id4 int ) distributed by('id3','id4');
```

Data file:

```
# cat x0.tbl
1|2|3|4|
0|0|0|0|
1|1|1|1|
```

```
2|2|2|2|
Loading process:
gbase> LOAD DATA INFILE ' sftp://gbase:gbase @192.168.105.66//opt/data/x0.tbl' INTO TABLE x0 DATA_ FORMAT 3 FIELDS TERMINATED BY '|';
Query OK, 4 rows affected (Elapsed: 00:00:01.11)
Task 4163 finished, Loaded 4 records, Skipped 0 records

Query inbound data:
gbase> select * from x0;
+----------+------+------+------+
| entry_ id | id2  | id3  | id4  |
+----------+------+------+------+
|        1 |    2 |    3 |    4 |
|        1 |    2 |    3 |    4 |
|        1 |    2 |    3 |    4 |
|        0 |    0 |    0 |    0 |
|        1 |    1 |    1 |    1 |
|        2 |    2 |    2 |    2 |
+----------+------+------+------+
6 rows in set (Elapsed: 00:00:00.03)
```

5.2.3 Query result export statement

5.2.3.1 Export query results

5.2.3.1.1 grammar

Function Description

GBase 8a MPP Cluster provides data export function, which exports data to the server side of the 8a cluster and also supports exporting cluster data to Hadoop or Kafka clusters. When exporting to 8a cluster servers and Hadoop clusters, it supports exporting to text files or gz/snap/lzo compressed files; Export to the Kafka cluster as a textual record.

SELECT... INTO OUTFILE... Support exporting query results for complex SQL statements. NameNode high availability is supported when exporting query results as HDFS files.

Grammar format

```
SELECT…INTO OUTFILE 'file_ path' [OUTFILE_OPTION] FROM…;
SELECT…FROM…INTO OUTFILE 'file_ path' [OUTFILE_OPTION];
```

Table 5154 Parameter Description

Field Name	Explanation of Meaning
file_ path	Save the path and file name of the exported data.
OUTFILE_ OPTION	Rules for data export.

Table 5155 OUTFILE_ Option parameter description

OUTFILE_ Option option	explain
FIELDS/COLUMNS TERMINATED BY	Field separator, supports multiple characters, with a maximum support of 10 characters. If no delimiter is specified, the default value is " t", which is the TAB key.
FIELDS/COLUMNS [OPTIONALLY] ENCLOSED BY	Field bounding box, you can specify a single character as the field bounding box, and specifying multiple characters will result in an error. Supports the OPTIONALLY option, which only applies to string types when added. Otherwise, it applies to all fields. The default is no field delimiter.
FIELDS/COLUMNS ESCAPED BY	Escape identifier. You can specify a single character as the escape identifier. If multiple characters are specified, an error will occur. The default value is "", and the default value is written as: FIELDS ESCAPED BY ' ' in the statement.

OUTFILE_ Option option	explain
LINES TERMINATED BY	Line separator, supports multiple characters, with a maximum support of 10 characters. The default is' n '.
LINES STARTING BY	Line start character, supports multiple characters, with a maximum support of 10 characters. The default is empty.
FIELDS/COLUMNS [OPTIONALLY] DOUBLE_ ENCLOSED BY	The field containing character is self escaped, which includes all the functions of the above parameter (field enclosing character); But the additional effect is that when the escape character is set to null, if a field meets the conditions for using a field delimiter and there are characters in the field that are the same as the field delimiter, the character will be self escaped through double writing.
NULL_ VALUE	A null value identifier that supports multiple characters and a maximum of 32 characters. The default is' N '.
OUTFILEMODE BY	Export method, optional values are: LOCAL or HDFS. Local: Export local files, HDFS: Export HDFS files. The default is to export in LOCAL mode.
WRITEMODE BY	Write method, optional values are NORMAL or OverWRITES. NORMAL: If the file already exists, an error will be reported, and OverWRITES will overwrite the existing file. The default is to write in NORMAL mode.
FILECOUNT	The number of files exported in parallel, with a minimum value of 0 and a maximum value of UINT_ MAX (4294967295), with a default value of 0, indicating no limit on the number of files exported. Only valid for exporting HDFS files. When the FILESIZE parameter is not specified, the actual number of exported HDFS files is the minimum of FILECOUNT and the number of data main shards. When using default values, each main shard is exported as an HDFS file.
FILESIZE	The maximum export file size, with a minimum value of 0 and a maximum value of ULONGLONG_ MAX (18446744073709551615), with a default value of 0, indicates that the size of the exported file is not limited. If the exported file size is greater than this parameter value, it will split to generate a new file. The new file naming method is file_ title+suffix+file_ Ext form, where file_ The title is file_ In name Previous section, file_ Ext is file_ In name The following section (including '.') suffix is an automatically appended file name suffix, with the first file name suffix being '_p1' and so on. The FILESIZE value supports the k/K/m/M/g/G suffix representation.
CHARACTER SET	Specify the character set for the exported file, support GBK and UTF8 encoding, and maintain consistency with the source table

OUTFILE_ Option option	explain
	character set by default.
FIELDS/COLUMNS LENGTH	The parameter used to set the field length when exporting using fixed length mode. When exporting fixed length format data, set the length of each field, separating multiple fields with commas. This parameter cannot be mixed with column separators and delimiters.
WITH HEAD	Optional parameter, when the user enters this parameter, it indicates that the local exported data file contains header information. When the user ignores this parameter, it indicates that the local exported data file does not contain header information. With HEAD syntax constraint: ● For the header export function, when the user specifies escape character, the header information will not be escaped; ● The default value for exporting header information is lowercase. If exporting headers requires case sensitivity, the parameter can be enabled:_ gcluster_ support_ outfile_ with_ table_ head_ case_ The default value for sensitive is 0, the function is turned off, and all exported headers are converted to lowercase; Set the value to 1, the function is enabled, and exporting header information is case sensitive ● The export of table headers only supports express engine tables, and other types are not guaranteed;

5.2.3.1.1.1 Export to HADOOP cluster description

Before executing the export statement, for HDFS that supports high availability, it is necessary to first set gbase_ hdfs_ namenodes='acitve_ nn, standby_ Nn ', specifies the highly available NameNode host information for HDFS. (HDFS typically consists of two NameNodes and several DataNodes, with one NameNode in the Active state and the other in the Standby state.)

The following example:

```
gbase> SET gbase_ hdfs_ namenodes="192.168.10.1,192.168.10.2";
```

Then the user enters an export statement, specifies the export of an HDFS file, and specifies the correct HDFS NameNode host name (or IP address) and port number in the URL.

```
gbase> SELECT * FROM test.t INTO OUTFILE ' hdp://hadoop
@192.168.10.1:50070/export/test.tbl' OUTFILEMODE BY HDFS;
```

When executing export statements, for supporting parallel export in multiple hdfs environments, it is necessary to assign namenodes of multiple hdfs environments to

gbase with '|' intervals_ hdfs_ namenodes。

gbase_ hdfs_ namenodes='hdfs1_ acitve_ nn, hdfs1_ standby_ nn | hdfs2_ acitve_ nn, hdfs2_ standby_ nn '

be careful

Perform HDFS export, configure/etc/hosts for all nodes in the cluster, and add IP address and host name mappings for Hadoop's Namenode and Datanode.

5.2.3.1.1.2 Export to Kafka Cluster Description

GBase 8a cluster produces query results data into all partitions or designated partitions of the kafka topic

select * from test.lineitem into outfile '
kafka://192.168.8.127:9092/test?brokers=192.168.8.127:9092
|192.168.8.127:9093&partition=0' [OPTIONS];

- KAFKA URL format:

kafka://broker1/topicname? [brokers=broker2|broker3|...][partition=pt1]

Brokers: A list of IP and port numbers for one or more broker addresses in the Kafka cluster, in the format host2: port2 | host3: port3 |. The 8a cluster will connect to the Kafka cluster through the broker address specified in the URL. Using the brokers parameter to specify multiple broker addresses can avoid not being able to connect to the Kafka cluster in the event of a single broker outage.

Topicname: specifies the name of the topic to be exported, which is case sensitive.

Partition: Specify a partition to export to the topic in the format of partition=ptnum. When this parameter is omitted, it is exported to all partitions of the topic by default.

- The supported Option parameters for exporting to kafka include:

Field delimiter, field delimiter, escape identifier, field delimiter self escaped (double_enclosed by), fixed length mode field length, long blob field text and base64 export method (fields/columns blobmode type_text/typebase64), line delimiter, line starting by, null value identifier, export file character set.

- The unsupported Option parameters are:

Export header information, export method (outfilemode by), write method (writeMode by overwrites), number of files exported in parallel (filecount), export file size, longblob field, independent file export method (files blobmode type_url)

- explain:

(1) Currently, parallel export of multiple nodes to Kafka is not supported. Messages are written to each broker in a polling manner, and the order of messages between multiple

brokers is not guaranteed during consumption.

(2) The Kafka cluster itself supports high availability. It is recommended to specify a topic partition with a replica count greater than 1 when configuring the Kafka cluster.

(3) If a broker is not available during the 8a cluster export process, the data will be automatically exported to other brokers to ensure high availability. Parameter gbase_ kafka_ producer_ message_ timeout_ The ms parameter specifies the timeout time for Kafka to send messages. If a message cannot be sent to the broker within the set time, an error message will be generated indicating a timeout. The default value is 300000 seconds, and the value range is 1~2147483647.

(4) Set gbase_ sql_ trace_ When level>=5, under the exported gnode, express.log will record the number of exported Kafka messages and the offset before and after the last message in all the topic partitions is exported.

(5) Support the configuration of the maximum number of bytes or the number of data rows contained in each Kafka message. The result set data of the 8a query will be divided into multiple Kafka messages based on this configuration:

Parameter gbase_ kafka_ producer_ message_ max_ Bytes is the maximum number of bytes in a Kafka message, with a default value of 8192 and a value range of 1~100000000

Parameter gbase_ kafka_ producer_ message_ max_ Rows is the maximum number of data rows in a Kafka message, with a default value of 100000000 and a value range of 1-4294967295

(6) Support the setting of the number of batch sent messages. When the efficiency of sending a single Kafka message is low, the number of batch sent messages can be configured. When a certain number of messages accumulate, they are sent together to the broker:

Parameter gbase_ kafka_ producer_ batch_ Messages is the number of messages sent in bulk, with a default value of 1000 and a value range of 1~10000000

(7) Kafka versions 0.11 and above support EOS features, and exporting cluster data to Kafka also supports EOS (exact once semantics) configuration, which does not duplicate production and consumption data and includes idempotence and transactional features. After enabling transactional configuration, idempotence configuration will be set to enable.

Parameter gbase_ kafka_ producer_ enable_ Idempotence is an idempotent switch, with a default value of 0 and a value range of 0 and 1. 0 represents turning off idempotence and 1 represents turning on idempotence. Enabling idempotence function can ensure that there are no duplicate messages generated by sending retry in the exported message (if the broker receives data but the ACK confirmation information returned to the sender is lost in the network, it will cause the sender to send duplicate messages again).

Parameter gbase_ kafka_ enable_ Transaction is a transactional switch, with a default value of 0 and values of 0 and 1. 0 represents turning off transactional activity and 1

represents turning on transactional activity. Enabling transactional functionality ensures that all exported messages belong to the same transaction, meaning that if one message fails to be produced, the entire transaction submission fails. After enabling transactional functions, only messages within successfully committed transactions can be imported when importing Kafka data into the 8a cluster.

(8) Export cluster data to Kafka and add two configuration parameters corresponding to the Kafka cluster:

Parameters_ gbase_ kafka_ producer_ transaction_ timeout_ Ms is the transaction timeout time, corresponding to transaction. timeout. ms in the Kafka configuration. The default value of this parameter is 600000, with a value range of 1000~2147483647.

Parameters_ gbase_ kafka_ producer_ queue_ buffering_ max_ Messages is the maximum number of messages in a batch, corresponding to queue. buffering. max. messages in the Kafka configuration, with a default value of 10000000 and a value range of 1~10000000

5.2.3.1.1.3 Export ORC file description

- The export syntax of the ORC file is the same as the regular export of 8a:

select ... into outfile 'file_ name' [option] from ...;

select ... from ... into outfile 'file_ name' [option];

Option parameter support:

1. Parameters that can be used normally: outfilemode by, writeMode by, filecount, filesize, character set

2. The syntax can be passed and executed normally, but it does not actually work. Warnings parameters will be reported: files/columns terminated by, files/columns enclosed by, files/columns captured by, lines terminated by, lines starting by, files/columns double_ enclosed by、null_ value、files/columns length、with head

- Orc file export supports local, ftp, sftp, and hdfs export methods
- Orc file export does not support remote export (rmt), kafka export, and http export
- Export of orc files requires specifying the export file name suffix as ". orc" or ". ORC". Exporting compressed orc files is not supported, such as exported text files with a suffix of. orc.gz that are still compressed
- The export of orc files supports configuring the exported orc file parameters: stripe size (default 64M), internal data compression format (none/zlib/zstd), and compressed block size (default 64K). This can be achieved by setting the configuration file, global, and session parameters, such as:

  set global gbase_ export_ orc_ stripe_ size=67108864

set global gbase_ export_ orc_ compression_ kind=zlib;

set global gbase_ export_ orc_ compression_ block_ size=65536

Note: Numbers are in bytes

- Export of ORC files supports setting the export ORC file size to exceed the limit of splitting. When the effective data length of the query result is specified by the parameter filesize to be greater than or equal to the filesize value, it will be split into a new file (data will be saved in rows, and new data files will not be truncated across rows). The default value of filesize is 0, which means there is no limit on the size of the exported file. The unit of filesize is bytes by default and supports K/M/G writing, such as 64M/16G
- ORC file export supports setting the number of files for parallel export of ORC files to HDFS, specified by the parameter filecount. By default, there is no limit on the number of files for parallel export, that is, each shard is exported in parallel as one file. If both the filecount and filesize parameters are specified, it means that filecount group files are exported in parallel, and each group of files is automatically split according to filesize. Only valid for exporting to HDFS.
- Orc file export supports setting the export file to automatically create a target directory, that is, to automatically create a target directory with the same name as the exported file during export. The parameter gbase can be used_ export_ Directory control, with a default value of 1 indicating automatic creation. When set to 0, a directory with the same name as the file name is not created. This parameter supports configuration file, global, and session settings.

5.2.3.1.2 export path

According to the export syntax, the SELECT... INTO OUTFILE... export path needs to be specified in the SQL statement, otherwise an error will be reported. The export path must be an absolute path and relative paths are not supported.

5.2.3.1.3 Export Method

- Automatic creation and file_ Name specifies a subdirectory with the same name as the file name, and the query results are exported as files with the same name in the subdirectory. By default (gbase_export_directory=1), this export method is recommended to ensure that the exported file is not confused with the names of existing files. When using this method, such as with file_ A subdirectory with the same name exists and is not empty. When WRITEMODE BY NORMAL is specified, an error will be reported to terminate the export; When WRITEMODE BY OVERWRITES is specified, it will automatically delete the file first_ Name all files in the subdirectory with the same name, and then export a new file.
- Do not create and file_ Name specifies a subdirectory with the same name as the file name, and the query results are exported as file_ The file specified by name

requires setting gbase for this method_ export_ The directory variable is 0;

The export path supports absolute paths, such as/home/gbase.

You can create a new folder under the "/home/gbase" directory to store files, such as/home/gbase/temp/test.txt.

explain

- Restrictions on the use of data export paths and files:
  - You must specify an export path.
  - If the current user does not have permission to create files in the export path, the system will report an error message

5.2.3.1.4 Fixed length export mode

premise

The maximum length of all fields is less than 16MB (16777216B).

The parameter definitions for the fixed length export mode are as follows:

Table 5156 Parameter Definition of Fixed Length Export Mode

Fixed length export conditions	Parameter settings
When the three specified parameters are set to null at the same time, it is a fixed length export	FIELDS TERMINATED BY '' ENCLOSED BY '' ESCAPED BY ''
When both specified parameters are set to empty values at the same time, it is a fixed length export	FIELDS TERMINATED BY '' ESCAPED BY ''
When the FIELDS/COLUMNS LENGTH parameter value is specified as the length of each field, it is exported as a fixed length.	fields length '11,5,9'

be careful

Export statements that use the fields length parameter cannot add or use the FIELDS Terminated BY '', ENCLOSED BY '', or ESCAPED BY '' parameters.

Example:

```
SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/length_ 1.txt' FIELDS TERMINATED BY '' ENCLOSED BY '' ESCAPED BY '';
```

Data export method

When exporting, each field is exported according to its maximum length. If the actual length of the field data is less than the maximum length of the field, spaces are filled in.

Escape Character Settings

In fixed length mode, it is mandatory to specify an escape character as empty, and the exported result is fixed length and does not escape any characters.

5.2.3.1.5 Escape character Export Mode

5.2.3.1.5.1 Determine whether the value of enclosed is true

When the following judgment conditions are met, the value of enclosed is TRUE:

1) If the field is of string type and a non empty delimiter is set through FIELDS ENCLOSED BY, then the value of ENCLOSED is TRUE.

2) If the field is of a non string type and a non empty delimiter is set through FIELDS ENCLOSED BY, the value of ENCLOSED is TRUE in the following two modes.

a) Specify an empty field separator and add the OPTIONALLY keyword to non empty delimiters, for example: FIELDS Terminated BY '' OPTIONALLY ENCLOSED BY ''.

b) Non empty parentheses without the OPTIONALLY keyword, for example: FIELDS ENCLOSED BY ''''.

5.2.3.1.5.2 Escape rules for character data

data type

Table -5157 Data Types

Character data type	
DATE	DATETIME
TIMESTAMP	TIME
CHAR	VARCHAR
BLOB	TEXT

Escape of data

If one of the following criteria is met, character x needs to be escaped:

- The character x is equal to the first character of the escape character.
- The character x is equal to the first character of the line separator (FIELDS Terminated BY).
- The character x is equal to ' 0'.
- The value of enclosed is TURE, and the character x is equal to the first character of the field enclosure set by FIELDS ENCLOSED BY.
- The value of enclosed is not TURE, and the character x is equal to the first character of the field separator set by FIELDS Terminated BY.

Please refer to the following instructions for the determination rules of enclosed values.

Explanation of Escape Rules

- Under normal circumstances, use the escape character defined by the FIELDS ESCAPED BY keyword to escape characters;
- If the value of the FIELDS ENCLOSED BY keyword is specified as one of "n, t, r, b, 0, Z, N" and the character is the same as the "field delimiter first character", use the character itself to escape itself.

  Tables and data used in the example:

```
DROP TABLE IF EXISTS t;
CREATE TABLE t(n int, v1 varchar(5), v2 varchar(8));
INSERT INTO t VALUES(102, 'ab', 'xmny');
```

  Example:

```
SELECT * FROM t INTO OUTFILE '/home/gbase/temp/1.txt' FIELDS ENCLOSED BY 'n';
```

  Viewing the export results, the character 'n' in 'xmny' has been escaped using its own:

```
$ cat 1.txt
n102n    nabn    nxmnnyn
```

- If the value of the specified FIELDS ENCLOSED BY keyword does not belong to

one of 'n, t, r, b, 0, Z, N', an escape character is used for escape.

Example:

```
SELECT * FROM t INTO OUTFILE '/home/gbase/temp/2.txt' FIELDS ENCLOSED BY 'm';
```

View the export results and use the default escape character "" to escape the character "m" in "xmny":

```
$ cat 2.txt
m102m	mabm	mx\mnym
```

5.2.3.1.5.3 Escape rules for non character data

Under normal circumstances, escape characters only escape character data, but in some special cases, non character data can also be escaped.

data type

Table 5158 Data Types

Non character data types	
TINYINT	INT
SMALLINT	MEDIUMINT
BIGINT	BOOL
FLOAT	DOUBLE
DECIMAL	YEAR

Escape of data

- If 'FIELDS ENCLOSED BY' is specified as one of the special characters'., 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, e,+, - ', non character data types will be escaped.

  remarks:

  There are two ways to specify the first character of the field bounding box as a special character:

  Directly specify a special character as the first character of the 'field delimiter', for example:

```
SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/3.txt' FIELDS ENCLOSED BY '0';
```

  Specify that the 'field delimiter' is empty and that the first character of the 'field delimiter' is a special character, such as:

```
SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/4.txt' FIELDS TERMINATED BY '0' ENCLOSED BY '';
```

- If the first character of the field bounding box is specified as one of "n, t, r, b, 0, Z, N", it can only be the character "0", and then exported using its own escape method.

Tables and data used in the example:

```
DROP TABLE IF EXISTS t;
CREATE TABLE t(n int, v1 varchar(5), v2 varchar(8));
INSERT INTO t VALUES(102, 'ab', 'xmny');
```

Example:

```
SELECT * FROM t INTO OUTFILE '/home/gbase/temp/5.txt' FIELDS ENCLOSED BY '0';
```

Viewing the export results, the '0' in non character data '102' is escaped using itself:

```
$ cat 5.txt
010020    0ab0      0xmny0
```

If the specified 'FIELDS ENCLOSED BY' does not belong to one of 'n, t, r, b, 0, Z, N', then the escape character is used for escape.

Example:

```
SELECT * FROM t INTO OUTFILE '/home/gbase/temp/6.txt' FIELDS ENCLOSED BY '2';
```

Viewing the export results, '2' in non character data '102' was escaped using the default escape character '':

```
$ cat 6.txt
210\22    2ab2      2xmny2
```

- Special case explanation: If the exported character contains " 0", the exported result is "escape character+0" (see example in section 5.2.3.1.6).

5.2.3.1.5.4 Rules for self escape of parentheses

Conditions for entering escape processing mode

Precondition: By forcing the escape character to be empty through FIELDS ESCAPED BY '', by DOUBLE_ The ENCLOSED BY keyword specifies a field bounding box that meets the rules for using bounding boxes.

Escape methods for characters

After entering this escape mode, if the field meets the rules for using parentheses (including normal fields and NULL values set through the NULL_VALUE parameter), all characters in the field that are the same as the parentheses are escaped using double writing. For example:

```
CREATE TABLE "aa" ("n" int(11) DEFAULT NULL,  "v" varchar(5) DEFAULT NULL);
Insert INTO aa Values (10, NULL), (11, 'data a'), (NULL, 'bbb'), (12, 'a "b" c');
```

```
gbase> select * from aa into outfile '/home/davies/out.txt' fields escaped by ''
terminated by '|' optionally double_ enclosed by '"' null_value 'gg"gg';
Query OK, 4 rows affected (Elapsed: 00:00:00.03)
```

The exported file is:

```
$ cat out.txt
10|"gg""gg"
11 | 'Data a'
gg"gg|"bbb"
12|"a""b""c"
```

It can be seen that due to the setting of the OPTIONALLY keyword, non string type data does not need to be enclosed. Therefore, even if there is a NULL value in the column, it is still passed through NULL_ The value is set to a non empty NULL value and will not double write or escape the bounding box in the data; Only fields that need to be enclosed will be escaped using double writing.

5.2.3.1.5.5 Two Special Situations Without Escape

In some special cases, the escape character will not work, in other words, the exported data will not be escaped.

Special situations where character data is not escaped

If the field delimiter (FIELDS ENCLOSED BY) is set to blank, the characters in the exported field are the same as the first character of the field delimiter (FIELDS Terminated BY), and the first character of the field delimiter (FIELDS Terminated BY) is one of "n, t, r, b, 0, Z, N". In this case, the exported characters are not escaped.

Tables and data used in the example:

```
DROP TABLE IF EXISTS t;
CREATE TABLE    t(n int, v1 varchar(5), v2 varchar(8));
INSERT INTO t VALUES(102, 'ab', 'xmny');
```

Example:

```
SELECT * FROM t INTO OUTFILE '/home/gbase/temp/unescaped_ 1.txt'
FIELDS TERMINATED BY 'n' ENCLOSED BY '';
```

To view the exported file:

```
$ cat unescaped_ 1.txt
102nabnxmny
```

The character 'n' in 'xmny' in the visible string is not escaped.

Special cases of non character data not escaping

If opt_ Enclosed is determined to be true (see the explanation section for the criteria). When a character in the exported data is the same as the first character of the field

separator set by FIELDS Terminated BY, and the first character of the field separator is one of "., 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, e,+, -", the exported character is not escaped in this case.

explain:

If one of the following criteria is met, opt_ Enclosed is true:

- TERMINATED BY ';' OPTIONALLY ENCLOSED BY '"';
- ENCLOSED BY ";
- OPTIONALLY ENCLOSED BY ";
- Do not write ENCLOSED BY clause.

Tables and data used in the example:

```
DROP TABLE IF EXISTS t;
CREATE TABLE t(n int, v1 varchar(5), v2 varchar(8));
INSERT INTO t VALUES(102, 'ab', 'xmny');
```

Example 1: Using 'FIELDS Terminated BY'2' OPTIONALLY ENCLOSED BY '', '2' in non character data '102' is not escaped.

```
gbase> SELECT * FROM t INTO OUTFILE '/home/gbase/temp/unescaped_
2.txt' FIELDS TERMINATED BY '2' OPTIONALLY ENCLOSED BY '"';
Query OK, 1 row affected
```

To view the exported file:

```
$ cat unescaped_ 2.txt
1022"ab"2"xmny"
```

Example 2: Using 'FIELDS Terminated BY'2' ENCLOSED BY '', '2' in non character data '102' is not escaped.

```
gbase> SELECT * FROM t INTO OUTFILE '/home/gbase/temp/unescaped_
3.txt' FIELDS TERMINATED BY '2' ENCLOSED BY '';
Query OK, 1 row affected
```

To view the exported file:

```
$ cat unescaped_ 3.txt
1022ab2xmny
```

Example 3: "FIELDS Terminated BY '2' OPTIONALLY ENCLOSED BY "," 2 "in non character data" 102 "is not escaped.

```
gbase> SELECT * FROM t INTO OUTFILE '/home/gbase/temp/unescaped_
4.txt' FIELDS TERMINATED BY '2' OPTIONALLY ENCLOSED BY '';
Query OK, 1 row affected
```

To view the exported file:

```
$ cat unescaped_ 4.txt
1022ab2xmny
```

Example 4: Using 'FIELDS Terminated BY' 2 ',' 2 'in non character data' 102 'is not escaped.

```
gbase> SELECT * FROM t INTO OUTFILE '/home/gbase/temp/unescaped_5.txt' FIELDS TERMINATED BY '2';
Query OK, 1 row affected
```

To view the exported file:

```
$ cat unescaped_ 5.txt
1022ab2xmny
```

5.2.3.1.6 Example

5.2.3.1.6.1 Incorrect writing method

Example

If a certain parameter is set repeatedly, the last one set shall prevail.

```
gbase> SELECT * FROM test.t INTO OUTFILE '/home/gbase/temp/d.txt' FIELDS TERMINATED BY '\t' TERMINATED BY ';';
Query OK, 1 row affected
```

To view the exported file:

```
$ cat d.txt
102; ab; xmny
```

Equivalent to

```
gbase> SELECT * FROM test.t INTO OUTFILE '/home/gbase/temp/e.txt' FIELDS TERMINATED BY ';';
Query OK, 1 row affected
```

To view the exported file:

```
$ cat e.txt
102; ab; xmny
```

5.2.3.1.6.2 Do not specify field delimiters

Example

Example 1: Export query results for complex SQL statements.

Tables and data used in the example:

```
CREATE TABLE t3 (color_type VARCHAR(20),color_ count INT, in_ date DATE);
INSERT INTO t3 (color_type,in_date,color_count) VALUES('black','2010-09-11',18),('black','2010-10-05',18), ('black','2010-10-13',31), ('blue','2010-09-21',23), ('blue','2010-09-30',15), ('blue','2010-10-11',62), ('red','2010-09-12',41), ('red','2010-10-01',12), ('red','2010-10-05',11);
```

Export SQL statement:

```
gbase> SELECT NVL(color_type,'') as color_ type_ Show, NVL (DECO
DE (color_type, NULL, f_YearMonth | |'Total ', NVL (f_YearMonth, col
or_type | |'Subtotal')), 'Total') AS f_ YearMonth_ show,SUM(color_count)
 FROM (SELECT color_type,DATE_FORMAT(in_date, '%Y-%m') as f_
 YearMonth,color_ count FROM t3) t GROUP BY CUBE(color_type,f_Y
earMonth) ORDER BY color_ type,f_ YearMonth INTO OUTFILE '/ho
me/gbase/temp/t3.txt';
Query OK, 12 rows affected
```

To view the exported file:

```
$ cat t3.txt
black      2010-09 18
black      2010-10 49
Black Subtotal 67
blue       2010-09 38
blue       2010-10 62
Blue subtotal 100
red        2010-09 41
red        2010-10 23
Red subtotal 64
2010-09 Total 97
2010-10 Total 134
Total 231
```

5.2.3.1.6.3 Specify Field Delimiter

Example

Example 1: Specify the field separator as a single character ';'.

Tables and data used in the example:

```
CREATE TABLE product (p_id INT, p_name VARCHAR(20), p_ desc
VARCHAR(100)) REPLICATED;
INSERT INTO product VALUES (1, 'qianzi', 'qianzi\\qianzi');
INSERT INTO product VALUES (2, 'bandeng', 'ban"deng');
```

Export the SQL statement, specifying the field separator as a single character ';':

```
gbase> SELECT * FROM product INTO OUTFILE
'/home/gbase/temp/product.txt' FIELDS TERMINATED BY ';';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat product.txt
1; qianzi; qianzi\\qianzi
```

```
2; bandeng; ban"deng
```

Example 2: Specify the field separator as a single character ','.

Tables and data used in the example:

```
CREATE TABLE t7(a INT, b DECIMAL, c FLOAT, d DATETIME);
INSERT INTO t7 VALUES(1,2,3.345,'2011-11-11
11:11:11'),(3,5,5.678,'2011-11-11 22:22:22');
```

Export the SQL statement, specifying the field separator as a single character ',':

```
gbase> SELECT * FROM t7 INTO OUTFILE '/home/gbase/t7.txt' FIELDS
TERMINATED BY ',';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat t7.txt
1,2,3.345,2011-11-11 11:11:11
3,5,5.678,2011-11-11 22:22:22
```

5.2.3.1.6.4 Specify Field Wrapper

Example

Example 1: Specify a single field bounding box as' @ '.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT
NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ b.txt'
FIELDS ENCLOSED BY '@';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat gs_ b.txt
@1@     @qwer@
@2@     @asdf@
```

Example 2: When there is no OPTIONALLY option, it affects all fields.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT
NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ c.txt'
FIELDS ENCLOSED BY '"';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat gs_ c.txt
"1"     "qwer"
"2"     "asdf"
```

Example 3: When specifying the OPTIONALLY option, it only affects string types.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT
NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ d.txt'
FIELDS OPTIONALLY ENCLOSED BY '"';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat   gs_ d.txt
1       "qwer"
2       "asdf"
```

5.2.3.1.6.5 Specify escape identifier

Example

Example 1: Specify a single escape identifier as' c '.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT
NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ e.txt'
FIELDS ESCAPED BY 'c';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat gs_ e.txt
1       qwer
```

```
2	asdf
```

Example 2: Specifying an escape identifier with multiple characters will result in an error.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ e.txt' FIELDS ESCAPED BY '6c@#';
ERROR 1149 (42000): FIELDS ESCAPED STRING must be only one character
```

5.2.3.1.6.6 Specify line breaks

Example

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs VALUES(3,'nihao');
INSERT INTO gs VALUES(4, 'GBase');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ f.txt' LINES TERMINATED BY '@#$';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat gs_ f.txt
3	nihao@#$4	GBase@#$
```

5.2.3.1.6.7 Specify line start delimiter

Example

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export an SQL statement, specifying multiple characters as line beginning

delimiters:

gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ g.txt' LINES STARTING BY '@#$';

Query OK, 2 rows affected

To view the exported file:

$ cat gs_ g.txt

@#$1 qwer

@#$2 asdf

5.2.3.1.6.8 Specify the self escape of the bounding box

After entering this escape mode, if the field meets the rules for using parentheses (including normal fields and NULL values set through the NULL_VALUE parameter), all characters in the field that are the same as the parentheses are escaped using double writing.

Example

Tables and data used in the example:

CREATE TABLE "aa" ("n" int(11) DEFAULT NULL, "v" varchar(5) DEFAULT NULL);
Insert INTO aa Values (10, NULL), (11, 'data a'), (NULL, 'bbb'), (12, 'a "b" c');

Export an SQL statement, specifying multiple characters as line beginning

delimiters:

gbase> select * from aa into outfile '/home/davies/out.txt' fields escaped by '' terminated by '|' double_ enclosed by '"' null_value 'gg"gg';

Query OK, 4 rows affected (Elapsed: 00:00:00.03)

To view the exported file:

$ cat out.txt

"10"|"gg""gg"

11 | Data a

"gg""gg"|"bbb"

"12"|"a""b""c"

It can be seen that after entering this mode, all ordinary fields that need to be surrounded (such as a "b" c) and those that need to be surrounded by NULL_ The NULL values set for the VALUE parameter (such as gg "gg) are all escaped using double writing.

5.2.3.1.6.9 Specify NULL_ Value parameter

When exporting non fixed length, NULL can be used_ The VALUE parameter specifies the null value identifier for the export.

be careful

This parameter only takes effect during non fixed length exports. During fixed length exports, all NULL values in the field are filled in with spaces based on the field width.

Example

Tables and data used in the example:

```
CREATE TABLE "gt" ("n" int(11) DEFAULT NULL,  "v" varchar(5) DEFAULT NULL);
INSERT INTO gt VALUES(10, NULL),(NULL, 'bb');
```

Export SQL statement, specify NULL_ Value is the 'aaaa' value:

```
gbase> select * from gt into outfile '/home/davies/a' null_ value 'aaaa';
Query OK, 2 rows affected (Elapsed: 00:00:00.00)
```

To view the exported file:

```
$ cat a
10      aaaa
aaaa    bb
```

5.2.3.1.6.10 Specify export method

In the SELECT INTO OUTFILE statement, the OUTFILEMODE parameter can be specified to specify the export method. LOCAL represents local export, HDFS represents Hadoop export. If the OUTFILEMODE parameter is not written, it defaults to local export.

Example

Example 1: Specify the export method LOCAL.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ g.txt' OUTFILEMODE BY LOCAL;
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat gs_ g.txt
    1qwer
    2asdf
```

Example 2: Specify the export method HDFS.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE 'HDP://192.168.153.32:50070/export/gs_ g.txt? user=gbase' OUTFILEMODE BY HDFS;
```

To view the exported file:

```
$ bin/hdfs dfs –cat /export/ gs_ g.txt
    1qwer
    2asdf
```

5.2.3.1.6.11 Specify write method

In the SELECT INTO OUTFILE statement, the WRITEMODE parameter can be specified to specify the file write method. When specifying NORMAL for export, if the target file already exists, an error will be reported to terminate the export task; When exporting as OverWRITES, export the target file in overwrite mode.

Example

Example 1: Specify the write method NORMAL for export.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ g.txt' WRITEMODE BY NORMAL;
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat gs_ g.txt
    1qwer
    2asdf
```

Example 2: Specify the write method OverWRITES for export.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
```

```
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ g.txt' WRITEMODE BY OVERWRITES;
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat gs_ g.txt
    1qwer
    2asdf
```

5.2.3.1.6.12 Specify the number of exported files

In the SELECT INTO OUTFILE statement, the FILECOUNT parameter can be specified to specify the number of files to be exported in parallel. The file naming method is file_ title+suffix+file_ Ext form, where file_ The title is file_ In name Previous section, file_ Ext is file_ In name The following parts (including '.') suffix are automatically appended file name suffixes, with the first file name suffix being '_1' and so on.

Example

Example 1: Specify the number of files to export as FILECOUNT (this parameter does not work when exporting locally).

Tables and data used in the example:

```
DROP TABLE IF EXISTS test;
CREATE TABLE test (ps_partkey bigint,ps_suppkey bigint,ps_availqty bigint,ps_supplycost decimal(15,2),ps_ comment varchar(200));

gbase>select * from test;
+------+------+------+--------+-----------------------------------+
| a    | b    | c    | d      | e                                 |
+------+------+------+--------+-----------------------------------
|    1 |    2 | 3325 | 771.64 | , even theodolites. regular       |
|    1 | 2502 | 8076 | 993.49 | ven ideas. quickly                |
|    1 | 5002 | 3956 | 337.09 | after the fluffily ironic         |
|    1 | 7502 | 4069 | 357.84 | al, regular dependencies          |
|    2 |    3 | 8895 | 378.49 | nic accounts. final accounts      |
|    2 | 2503 | 4969 | 915.27 | ptotes. quickly pending           |
|    2 | 5003 | 8539 | 438.37 | blithely bold ideas. furiously    |
|    2 | 7503 | 3025 | 306.39 | olites. deposits wake carefully   |
|    3 |    4 | 4651 | 920.92 | ilent foxes affix furiously quickly |
|    3 | 2504 | 4093 | 498.13 | ending dependencies haggle fluffily |
|    3 | 5004 | 3917 | 645.40 | of the blithely regular theodolites |
```

```
|       3 | 7504 | 9942 | 191.92 | unusual, ironic foxes according          |
|       4 |       5 | 1339 | 113.97 | carefully unusual ideas. packages      |
|       4 | 2505 | 6377 | 591.18 | ly final courts haggle                          |
|       4 | 5005 | 2694 |    51.37 | g, regular deposits: quick                 |
+------+------+------+--------+-----------------------------------
800000 rows in set
```

Export SQL statement:

```
gbase> SELECT * FROM test INTO OUTFILE 'HDP://192.168.153.21:50070/export/test.txt? user=gbase' OUTFILEMODE BY HDFS FILECOUNT 3;
Query OK, 800000 rows affected
```

To view the exported file:

```
$ bin/hdfs dfs –ls /export
test_ 1.txt
test_ 2.txt
test_ 3.txt
```

5.2.3.1.6.13 Specify export file size

Example

Example 1: Specify the export file size FILESIZE.

Tables and data used in the example:

```
DROP TABLE IF EXISTS test;
CREATE TABLE test (ps_partkey bigint,ps_suppkey bigint,ps_availqty bigint,ps_supplycost decimal(15,2),ps_ comment varchar(200));

gbase> select * from test;
+------+------+------+--------+-----------------------------------+
| a       | b       | c       | d           | e                                                  |
+------+------+------+--------+-----------------------------------
|       1 |       2 | 3325 | 771.64 | , even theodolites. regular              |
|       1 | 2502 | 8076 | 993.49 | ven ideas. quickly                          |
|       1 | 5002 | 3956 | 337.09 | after the fluffily ironic                  |
|       1 | 7502 | 4069 | 357.84 | al, regular dependencies                   |
|       2 |       3 | 8895 | 378.49 | nic accounts. final accounts             |
|       2 | 2503 | 4969 | 915.27 | ptotes. quickly pending                    |
|       2 | 5003 | 8539 | 438.37 | blithely bold ideas. furiously          |
|       2 | 7503 | 3025 | 306.39 | olites. deposits wake carefully         |
|       3 |       4 | 4651 | 920.92 | ilent foxes affix furiously quickly |
|       3 | 2504 | 4093 | 498.13 | ending dependencies haggle fluffily |
|       3 | 5004 | 3917 | 645.40 | of the blithely regular theodolites |
|       3 | 7504 | 9942 | 191.92 | unusual, ironic foxes according         |
|       4 |       5 | 1339 | 113.97 | carefully unusual ideas. packages      |
```

```
|      4 | 2505 | 6377 | 591.18 | ly final courts haggle                          |
|      4 | 5005 | 2694 |   51.37 | g, regular deposits: quick                      |
+------+------+------+--------+-----------------------------------
800000 rows in set
```

Export SQL statement:

```
gbase> SELECT * FROM test INTO OUTFILE '/home/gbase/temp/test.txt'
FILESIZE 33554432;
Query OK, 800000 rows affected
```

To view the exported file:

```
$ ll
test_ p1.txt
test_ p2.txt
test_ p3.txt
```

5.2.3.1.6.14 Export compressed format files

Example

Example 1: Export a. gz format file.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT
NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ g.gz'
WRITEMODE BY NORMAL;
Query OK, 2 rows affected
```

To view the exported file:

```
$ ll
gs_ g.gz
```

Example 2: Export a. snappy format file.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT
NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_
g.snappy' WRITEMODE BY NORMAL;
Query OK, 2 rows affected
```

To view the exported file:

```
$ ll
gs_ g.snappy
```

Example 3: Export a. lzo format file.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs values(1,'qwer'),(2,'asdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs_ g.lzo' WRITEMODE BY NORMAL;
Query OK, 2 rows affected
```

To view the exported file:

```
$ ll
gs_ g.lzo
```

5.2.3.1.6.15 Export data with header

Example

Tables and data used in the example:

```
DROP TABLE IF EXISTS test;
CREATE TABLE test (ps_partkey bigint,ps_suppkey bigint,ps_availqty bigint,ps_supplycost decimal(15,2),ps_ comment varchar(200));

gbase> select * from test;
+------+------+------+--------+-----------------------------------+
|ps_ partkey|ps_ suppkey|ps_ availqty|ps_ supplycost|ps_ comment              |
+------+------+------+--------+-----------------------------------
|       1 |       2 | 3325 | 771.64 | , even theodolites. regular              |
|       1 | 2502 | 8076 | 993.49 | ven ideas. quickly                        |
|       1 | 5002 | 3956 | 337.09 | after the fluffily ironic                 |
|       1 | 7502 | 4069 | 357.84 | al, regular dependencies                  |
|       2 |       3 | 8895 | 378.49 | nic accounts. final accounts           |
|       2 | 2503 | 4969 | 915.27 | ptotes. quickly pending                   |
|       2 | 5003 | 8539 | 438.37 | blithely bold ideas. furiously         |
|       2 | 7503 | 3025 | 306.39 | olites. deposits wake carefully        |
|       3 |       4 | 4651 | 920.92 | ilent foxes affix furiously quickly |
|       3 | 2504 | 4093 | 498.13 | ending dependencies haggle fluffily |
|       3 | 5004 | 3917 | 645.40 | of the blithely regular theodolites |
|       3 | 7504 | 9942 | 191.92 | unusual, ironic foxes according        |
|       4 |       5 | 1339 | 113.97 | carefully unusual ideas. packages      |
```

```
|      4 | 2505 | 6377 | 591.18 | ly final courts haggle                    |
|      4 | 5005 | 2694 |   51.37 | g, regular deposits: quick                |
+------+------+------+--------+-------------------------------------
800000 rows in set
```

Export SQL statement:

```
gbase> SELECT * FROM test INTO OUTFILE '/home/gbase/temp/test.tbl'
fields terminated by '|' with head;
Query OK, 800000 rows affected
```

To view the exported file:

```
$ cat test.tbl
ps_ partkey| ps_ suppkey| ps_ availqty| ps_ supplycost| ps_ comment
1|2|3325|771.64|, even theodolites. Regular
1|2502|8076|993.49|ven ideas. quickly
1|5002|3956|337.09|after the fluffily ironic
.....
.....
```

5.2.3.1.6.16 Export header is case sensitive

Example

set _ gcluster_ support_ outfile_ with_ table_ head_ case_ sensitive=1;

The default value is 0, the function is turned off, and all exported headers are converted to lowercase; Set the value to 1, the function is enabled, and exporting header information is case sensitive

```
gbase> select * from tb;
+------+------+------+
| A    | b    | D    |
+------+------+------+
|    1 | aaa  | AAA  |
+------+------+------+
```

1. Select does not specify column names, and the exported header information matches the case of the column names in the original data table

rmt:select * from tb into outfile 'path' with head;

```
AbD
1 aaa AAA
```

2. Select specifies the column name, and the capitalization of the exported header information is consistent with the writing method in the exported SQL

rmt:select a,B,d from tb into outfile 'path' with head;

```
aBd
1aaaAAA
```

3. The select statement uses as to specify an alias for the column, and the casing of the exported header information is consistent with the alias set by as

rmt:select a as BIG_ A,b as BIG_ B,d as BIG_ D from tb into outfile 'path'

```
with head;
        BIG_    ABIG_    BBIG_ D
        1       aaa       AAA
```

5.2.3.1.6.17 Export data using the CHARACTER SET parameter

Example

Tables and data used in the example:

```
CREATE TABLE "test_3" (  "column_1" int(11) DEFAULT NULL,
"column_2" varchar(10) DEFAULT NULL,  "column_3" varchar(20)
DEFAULT NULL);

gbase> select * from test_ 3;
+----------+----------+----------+
| column_ 1 | column_ 2 | column_ 3 |
+----------+----------+----------+
|     59432 | gg          | laqpqpd  |
|     43452 | sisoekso | mozoa,a  |
|      3890 | lqps,rpd | gg          |
+----------+----------+----------+
3 rows in set (Elapsed: 00:00:00.01)
```

Export SQL statement:

```
gbase> select * from test_ 3 into outfile '/home/gbase/test.txt'   character set
gbk;
Query OK, 3 rows affected (Elapsed: 00:00:00.29)
```

To view the exported file:

```
$cat test.txt:
43452   sisoekso        mozoa,a
3890    lqps,rpd        gg
59432   gg      laqpqpd
```

5.2.3.1.6.18 Export data using the FIELDS/COLUMNS LENGTH parameter

Example

Tables and data used in the example:

```
CREATE TABLE "test_2" (  "column_1" int(11) DEFAULT NULL,
"column_2" varchar(10) DEFAULT NULL,  "column_3" varchar(20)
DEFAULT NULL);
```

```
gbase> select * from test_ 2;
+----------+----------+----------+
| column_ 1 | column_ 2 | column_ 3 |
+----------+----------+----------+
|       59432 | gg             | laqpqpd    |
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | gg             |
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | gg             |
|       59432 | gg             | laqpqpd    |
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | gg             |
|       59432 | gg             | laqpqpd    |
|       59432 | gg             | laqpqpd    |
|       59432 | gg             | laqpqpd    |
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | gg             |
|       59432 | gg             | laqpqpd    |
|       59432 | gg             | laqpqpd    |
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | gg             |
|       59432 | gg             | laqpqpd    |
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | gg             |
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | gg             |
|       43452 | sisoekso | mozoa,a    |
|         3890 | lqps,rpd | gg             |
+----------+----------+----------+
24 rows in set (Elapsed: 00:00:00.01)
```

Export SQL statement:

```
gbase> select * from test_ 2 into outfile '/home/gbase/test.txt' fields length '11,5,9';
Query OK, 24 rows affected (Elapsed: 00:00:00.29)
```

To view the exported file:

```
$cat   test.txt
43452       sisoemozoa,a
3890        lqps,gg
59432       gg     laqpqpd
59432       gg     laqpqpd
43452       sisoemozoa,a
3890        lqps,gg
59432       gg     laqpqpd
43452       sisoemozoa,a
```

```
3890    lqps,gg
43452   sisoemozoa,a
3890    lqps,gg
43452   sisoemozoa,a
3890    lqps,gg
59432   gg      laqpqpd
43452   sisoemozoa,a
3890    lqps,gg
43452   sisoemozoa,a
3890    lqps,gg
59432   gg      laqpqpd
43452   sisoemozoa,a
3890    lqps,gg
59432   gg      laqpqpd
59432   gg      laqpqpd
59432   gg      laqpqpd
```

5.2.3.1.7 Special examples

5.2.3.1.7.1 Processing of data containing NULL values

If the content of a field in the data to be exported is a NULL value, the exported NULL text for that field is "current escape character+N".

The default escape character is "", so the exported NULL text for the field is " N".

Example

Example 1: If the escape character defaults to '', the exported result of the 'NULL' value is' N '.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs VALUES(NULL,NULL);
INSERT INTO gs VALUES(1, 'GBase');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/null_ 1.txt';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat null_ 1.txt
\N      \N
1       GBase
```

Example 2: If a field delimiter is specified in the export statement, it has no effect on NULL values.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs VALUES(NULL,NULL);
INSERT INTO gs VALUES(1, 'GBase');
```

Export SQL statement:

```
gbase>  SELECT * FROM gs INTO OUTFILE '/home/gbase/null_ 2.txt' FIELDS ENCLOSED BY '"';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat null_ 2.txt
\N      \N
"1"     "GBase"
```

Example 3: If the escape character is set to '|', the exported result of the 'NULL' value is' | N '.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs VALUES(NULL,NULL);
INSERT INTO gs VALUES(1, 'GBase');
```

Export SQL statement:

```
gbase>  SELECT * FROM gs INTO OUTFILE '/home/gbase/null_ 3.txt' FIELDS ESCAPED BY '|';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat null_ 3.txt
|N      |N
1       GBase
```

5.2.3.1.7.2 Handling Escape Characters as Empty Characters

Example

Example 1: If the character in FIELDS ESCAPED BY '' is a null character, then NULL is output as NULL, not as' N '.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs VALUES(NULL,NULL);
INSERT INTO gs VALUES(1, 'GBase');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/esp_ 1.txt' FIELDS ESCAPED BY '';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat esp_ 1.txt
NULL    NULL
GBase
```

Example 2: If a field delimiter is specified in the export statement, it still has no effect on NULL values.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs VALUES(NULL,NULL);
INSERT INTO gs VALUES(1, 'GBase');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/esp_ 2.txt' FIELDS ESCAPED BY '' ENCLOSED BY '"';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat esp_ 2.txt
NULL    NULL
"1"     "GBase"
```

5.2.3.1.7.3 Processing of data containing ' 0' characters

Example

If the value of a field (usually a string type, such as varchar) in the exported data is " 0", by default, the character in the exported text is " 0".

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs VALUES(3,'asdf\0dv');
INSERT INTO gs VALUES(4, 'GBase');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/test_ 1.txt';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat test_ 1.txt
3	asdf\0dv
4	GBase
```

5.2.3.1.7.4 Handling when specifying multiple characters as field separators and the text also contains multiple character separators

Example

If multiple characters are specified as delimiters in the 'SELECT INTO OUTFILE' statement, and the field text contains the delimiter string, only the first character of the delimiter string is escaped.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs VALUES(3,'nihao');
INSERT INTO gs VALUES(4, 'GBase');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/gs.txt' FIELDS TERMINATED BY 'ih';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat gs.txt
3ihn\ihao
4ihGBase
```

5.2.3.1.7.5 Handling when the field text contains " n" or " r"

Example

If a field (usually of string type, such as varchar) in the exported data contains " n" or " r", only " n" is escaped.

If a escape character is added before " n" (the default is ""), " r" remains unchanged, but it is still an invisible character " r", which is viewed as "0x0D" in binary mode.

Example 1: The reason for escaping ' n' is because the content ' n' in the text is the same as the default line delimiter (Lines Terminated), so the ' n' in the text is escaped.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs values(1,'qw\ner'),(2,'as\rdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/n_ 1.txt';
Query OK, 2 rows affected
```

To view the exported file, the - b parameter in the cat command indicates numbering non empty output lines:

```
$ cat -b n_ 1.txt
     1  1          qw\
     2  er
df   3  2          as
```

To view exported files using binary mode:

```
$ hexdump -C n_ 1.txt
00000000  31 09 71 77 5c 0a 65 72  0a 32 09 61 73 0d 64 66  |1.qw\.er.2.as.df|
00000010  0a                                                |.|
00000011
```

Example 2: If the specified line separator displayed is a different character, no escape occurs.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(20) DEFAULT NULL);
INSERT INTO gs values(1,'qw\ner'),(2,'as\rdf');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/n_ 2.txt'
LINES TERMINATED BY ';';
```

View the exported file, where '^ M' represents' n ':

```
$ vi n_ 2.txt
1          qw
er; 2      as^Mdf;
```

To view exported files using binary mode:

```
$ hexdump -C n_ 2.txt
00000000  31 09 71 77 0a 65 72 3b  32 09 61 73 0d 64 66 3b  |1.qw.er; 2.as.df;|
```

```
00000010
```

5.2.3.1.7.6 Export of fixed length mode

Example

Export data in fixed length mode.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(25) DEFAULT NULL);
INSERT INTO gs values(1,'GBase 8a'),(2,'GBase 8a MPP Cluster');
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/length_ 1.txt' FIELDS TERMINATED BY '' ENCLOSED BY '' ESCAPED BY '';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat length_ 1.txt
1               GBase 8a
2               GBase 8a MPP Cluster
```

Using binary mode to view the exported file, using space completion when exporting data:

```
$ hexdump -C length_ 1.txt
00000000  31 20 20 20 20 20 20 20  20 20 20 47 42 61 73 65  |1       GBase|
00000010  20 38 61 20 20 20 20 20  20 20 20 20 20 20 20 20  | 8a              |
00000020  20 20 20 20 20 20 20 20  20 20 20 20 20 20 20 20  |                 |
*
00000050  20 20 20 20 20 20 0a 32  20 20 20 20 20 20 20 20  |      .2         |
00000060  20 20 47 42 61 73 65 20  38 61 20 4d 50 50 20 43  |  GBase 8a MPP C|
00000070  6c 75 73 74 65 72 20 20  20 20 20 20 20 20 20 20  |luster          |
00000080  20 20 20 20 20 20 20 20  20 20 20 20 20 20 20 20  |                 |
*
000000a0  20 20 20 20 20 20 20 20  20 20 20 20 20 20 0a     |              .|
000000ae
```

be careful

- A varchar character may occupy multiple bytes. When the current character set is utf8, a varchar character occupies three bytes. If varchar (10) is set during table creation, the maximum length of this field may occupy 30 bytes.

5.2.3.1.7.7 Export of Fixed Length Patterns with NULL Values

Example

Example 1: Fixed length mode for exporting null values: Fill in all fields with spaces based on their width.

If the value inserted for the integer field a is NULL, the actual length used is 0, and exporting using fixed length mode will use 11 spaces to fill in.

Tables and data used in the example:

```
DROP TABLE IF EXISTS gs;
CREATE TABLE gs (a int DEFAULT NULL, b varchar(25) DEFAULT NULL);
INSERT INTO gs values(NULL,'GBase 8a'),(NULL,NULL);
```

Export SQL statement:

```
gbase> SELECT * FROM gs INTO OUTFILE '/home/gbase/temp/length_2.txt' FIELDS TERMINATED BY '' ENCLOSED BY '' ESCAPED BY '';
Query OK, 2 rows affected
```

To view the exported file:

```
$ cat length_ 2.txt
                GBase 8a
```

To view exported files using binary mode:

```
$ hexdump -C length_ 2.txt
00000000  20 20 20 20 20 20 20 20  20 20 20 47 42 61 73 65  |           GBase|
00000010  20 38 61 20 20 20 20 20  20 20 20 20 20 20 20 20  | 8a             |
00000020  20 20 20 20 20 20 20 20  20 20 20 20 20 20 20 20  |                |
*
00000050  20 20 20 20 20 20 0a 20  20 20 20 20 20 20 20 20  |      .         |
00000060  20 20 20 20 20 20 20 20  20 20 20 20 20 20 20 20  |                |
*
```

```
000000a0  20  20  20  20  20  20  20  20    20  20  20  20  20  0a
|                .|
000000ae
```

5.2.3.2 Remote export of query results

5.2.3.2.1 grammar

Function Description

GBase 8a MPP Cluster provides remote data export function, which exports data from the cluster server to the machine where the cluster client is located, and the exported data is a text file.

Grammar format

rmt:select_ syntax INTO OUTFILE '*file_ path*' [*OUTFILE_OPTION*];

Table 5159 Parameter Description

Field Name	Explanation of Meaning
file_ path	Save the path and file name of the exported data.
OUTFILE_ OPTION	Rules for data export.

Table 5160 OUTFILE_ Option parameter description

Field Name	Explanation of Meaning
FIELDS/COLUMNS TERMINATED BY	Field separator, supporting multiple characters. If no separator is specified, the default value is " t", which is the TAB key.
FIELDS/COLUMNS [OPTIONALLY] ENCLOSED BY	Field bounding box, you can specify a single character as the field bounding box, and specifying multiple characters will result in an error. Supports the OPTIONALLY option, which only applies to string types when added. Otherwise, it applies to all fields. The default is no field delimiter.
FIELDS/COLUMNS ESCAPED BY	Escape identifier. You can specify a single character as the escape identifier. If multiple characters are specified, an error will occur. The default value is "", and the default value is written as: FIELDS ESCAPED BY ' ' in the statement.
LINES TERMINATED BY	Line separator, supports multiple characters, defaults to ' n'.
LINES STARTING BY	Line start character, supports multiple characters, defaults to '', which is empty.
FIELDS/COLUMNS [OPTIONALLY]	The field containing character is self escaped, which includes all the functions of the above parameter (field enclosing

Field Name	Explanation of Meaning
DOUBLE_ ENCLOSED BY	character); But the additional effect is that when the escape character is set to null, if a field meets the conditions for using a field delimiter and there are characters in the field that are the same as the field delimiter, the character will be self escaped through double writing.
NULL_ VALUE	A null value identifier that supports multiple characters and defaults to ' N'.
CHARACTER SET	Specify the character set for the exported file, support GBK and UTF8 encoding, and do not write. The default export and source table character set should be consistent.
FIELDS/COLUMNS LENGTH	The parameter used to set the field length when exporting using fixed length mode. When exporting fixed length format data, set the length of each field, separating multiple fields with commas. This parameter cannot be mixed with column separators and delimiters.
WITH HEAD	With HEAD is an optional parameter that, when entered by the user, indicates that the data file is exported remotely with header information. When the user ignores this parameter, it indicates that there is no header information when exporting data files remotely, which is consistent with the behavior of existing export tools. With HEAD syntax constraint: ● For the header export function, when the user specifies escape character, the header information will not be escaped; ● The export of table headers only supports express engine tables, and other types are not guaranteed;

be careful

- Rmt: and select_ There cannot be spaces between syntax.
- file_ Path, and field_ The field delimiters, field delimiters, escape identifiers, etc. in the Option must be surrounded by single quotation marks "", otherwise the data cannot be exported normally.
- Remote export does not support top-level queries that are UNION class queries.
- If the target file for remote export already exists locally, it cannot be selected @ @ error_ Count to obtain the number of errors, because the existence of the file is determined by the client, and @ @ error_ Count is a server-side variable.

5.2.3.2.2 export path

According to the export syntax, the remote export path needs to be specified in the SQL statement, otherwise an error will be reported. However, after specifying the export path, the creation of files can also be divided into the following situations:

- If specified as an absolute path, the exported file will be in that path;
- If specified as a relative path, the exported file path is "the directory where the user logs in to the client"+the relative path set in the SQL statement. If the directory where the user logs in to the client is/opt, then use the select into outfile statement to export: select * from t into outfile 'test/1' fields terminated by ';'; At this point, the complete path of the exported file is:/opt/test/1

5.2.3.2.3 Export Port

The remote export function uses the 16066-16165 port range of the machine where the export client is located as the ephemeral port of the export service by default. The port range can be configured by modifying the configuration file parameters, as shown in the table below.

Table 5161 Parameter Description

name	minimum value	Maximum value	Default value
remote_ export_ min_ port	one thousand and twenty-five	sixty-five thousand five hundred and thirty-five	sixteen thousand and sixty-six
remote_ export_ max_ port	one thousand and twenty-five	sixty-five thousand five hundred and thirty-five	sixteen thousand one hundred and sixty-six

Before using the remote export function, please confirm that the port range is open normally and not occupied, otherwise it may cause remote export failure.

5.2.3.2.4 Deploy Remote Export Client

Preparation before deployment

Firstly, it is necessary to create a DBA (using gbase users as an example) user on the remote export client server.

Steps for Deploying a Remote Export Client

Refer to Chapter 3.2.5 for deployment methods to install the client

Remote client login cluster service

Example: As a GBase user, log in to cluster node 192.168.10.115.

```
$ gccli -ugbase -pgbase -h192.168.10.115
```

Table -5162 Parameter Description

Field Name	Explanation of Meaning
-u	Login username.
-p	Login user password.
-h	Logged in cluster node IP.

5.2.3.2.5 Example

This section only lists some usage examples. For other examples, refer to the "Query Result Export Examples" section. When exporting, add rmt: before the SQL statement.

5.2.3.2.5.1 Do not specify field delimiters

Example

Tables and data used in the example:

```
DROP TABLE IF EXISTS cust;
CREATE TABLE cust(c_id INT, c_name VARCHAR(20), c_ addr VARCHAR(100));
INSERT INTO cust VALUES (1, 'xiaoming', 'Tianjin');
INSERT INTO cust VALUES (3, 'qiaorui', 'Hebei');
INSERT INTO cust VALUES (4, 'tianfei', 'Anhui');
INSERT INTO cust VALUES (2, 'zhangling', 'Hunan');
```

Export SQL statements without specifying field delimiters, i.e. using the default field delimiter ' t':

```
gbase> rmt:SELECT * FROM cust INTO OUTFILE '/home/gbase/temp/cust.txt';
Query OK, 4 rows affected
```

To view the exported file:

```
$ cat cust.txt
1       xiaoming        Tianjin
3       qiaorui Hebei
4       tianfei Anhui
2       zhangling       Hunan
```

5.2.3.2.5.2 Specify Field Delimiter

Example

Tables and data used in the example:

```
DROP TABLE IF EXISTS cust;
```

```
CREATE TABLE cust(c_id INT, c_name VARCHAR(20), c_ addr VARCHAR(100));
INSERT INTO cust VALUES (1, 'xiaoming', 'Tianjin');
INSERT INTO cust VALUES (3, 'qiaorui', 'Hebei');
INSERT INTO cust VALUES (4, 'tianfei', 'Anhui');
INSERT INTO cust VALUES (2, 'zhangling', 'Hunan');
```

Export the SQL statement, specifying the field separator as', ':

```
gbase> rmt:SELECT * FROM cust INTO OUTFILE '/home/gbase/temp/cust.txt' FIELDS TERMINATED BY ',';
```

Query OK, 4 rows affected

To view the exported file:

To view the exported file:

```
$ cat cust.txt
1,xiaoming,Tianjin
3,qiaorui,Hebei
4,tianfei,Anhui
2,zhangling,Hunan
```

Export the SQL statement, specifying the field separator as'; ':

```
gbase> rmt:SELECT * FROM cust INTO OUTFILE '/home/gbase/temp/cust.txt' FIELDS TERMINATED BY ';';
```

Query OK, 4 rows affected

To view the exported file:

```
$ cat cust.txt
1; xiaoming; Tianjin
3; qiaorui; Hebei
4; tianfei; Anhui
2; zhangling; Hunan
```

5.2.3.2.5.3 Specify the field delimiter as' '

Example

Tables and data used in the example:

```
DROP TABLE IF EXISTS cust;
CREATE TABLE cust(c_id INT, c_name VARCHAR(20), c_ addr VARCHAR(100));
INSERT INTO cust VALUES (1, 'xiaoming', 'Tianjin');
INSERT INTO cust VALUES (3, 'qiaorui', 'Hebei');
INSERT INTO cust VALUES (4, 'tianfei', 'Anhui');
INSERT INTO cust VALUES (2, 'zhangling', 'Hunan');
```

Export SQL statement:

```
gbase> rmt:SELECT * FROM cust INTO OUTFILE '/home/gbase/temp/cust.txt' FIELDS TERMINATED BY ';' ENCLOSED BY
```

```
"";
Query OK, 4 rows affected
To view the exported file:
$ cat cust.txt
"1"; "xiaoming"; "Tianjin"
"3"; "qiaorui"; "Hebei"
"4"; "tianfei"; "Anhui"
"2"; "zhangling"; "Hunan"
```

5.2.3.2.5.4 Specify the escape character as' g '

Example

```
Tables and data used in the example:
DROP TABLE IF EXISTS product;
CREATE TABLE product (p_id INT, p_name VARCHAR(20), p_ desc VARCHAR(100));
INSERT INTO product VALUES (1, 'qianzi', 'qianzi\\qianzi');
INSERT INTO product VALUES (2, 'bandeng', 'ban"deng');
INSERT INTO product VALUES (4, 'jiandao', 'Hei;bei');
INSERT INTO product VALUES (3, 'chazi', 'Anh\nui');
INSERT INTO product VALUES (5, 'canzhuo', 'Hunan');

gbase> SELECT * FROM product;
+------+---------+---------------+
| p_ id | p_ name    | p_ desc              |
+------+---------+---------------+
|       1 | qianzi    | qianzi\qianzi |
|       2 | bandeng | ban"deng           |
|       4 | jiandao | Hei; bei             |
|       3 | chazi      | Anh
ui               |
|       5 | canzhuo | Hunan                |
+------+---------+---------------+
5 rows in set
Export SQL statement:
gbase> rmt:SELECT * FROM product INTO OUTFILE '/home/gbase/temp/product.txt' FIELDS TERMINATED BY ';' ESCAPED BY 'g';
Query OK, 5 rows affected
To view the exported file:
$ cat product.txt
1; qianzi; qianzi\qianzi
2; bandengg; ban"dengg
```

```
4; jiandao; Heig; bei
3; chazi; Anhg
ui
5; canzhuo; Hunan
```

explain

- "" is not set to "g" because "" is no longer considered as a special character after specifying other characters as escape character.
- Both ";" and " n" are set as escape characters "g" to represent data, rather than the field delimiter ";" and row delimiter " n".
- 'g' is also preceded by the escape character 'g' because 'g' is treated as a special character after being specified as an escape character.

5.2.3.2.5.5 Use Cases in Precautions

Example

Example 1: "rmt:" and select_ There cannot be spaces between syntax.

```
Statements that can correctly export data:
rmt:SELECT * FROM cust INTO OUTFILE '/home/gbase/temp/cust.txt';
Statements that will report syntax error:
rmt: SELECT * FROM cust INTO OUTFILE '/home/gbase/temp/cust.txt';
```

Example 2: file_ Path, and field_ The field delimiters, field delimiters, escape identifiers, etc. in the Option must be surrounded by single quotation marks """, otherwise the data cannot be exported normally.

```
Statements that can correctly export data:
rmt:SELECT * FROM cust INTO OUTFILE '/home/gbase/temp/cust.txt';
Statements that will report syntax error:
rmt:SELECT * FROM cust INTO OUTFILE /home/gbase/temp/cust.txt;
```

Example 3: Remote export does not support top-level queries that are UNION class queries.

```
Statements that will report syntax error:
rmt:SELECT * FROM cust UNION SELECT * FROM product INTO OUTFILE '/home/gbase/temp/product.txt';
ERROR 1149 (42000): (GBA-02SC-1001) SELECT INTO OUTFILE with UNION is not supported.
```

Example 4: If the target file for remote export already exists locally, it cannot be selected @ @ error_ Count to obtain the number of errors, because the existence of the file is determined by the client, and @ @ error_ Count is a server-side variable.

Tables and data used in the example:

```
create table t(id int);
insert into t values(1);
```

Export SQL statement:

```
gbase> rmt:select * from t into outfile '/home/gbase/t.txt';
Query OK, 1 row affected
gbase> rmt:select * from t into outfile '/home/gbase/t.txt';
ERROR:
Can't open file '/home/gbase/t.txt' to write. Caused by: File exists
gbase> select @@error_ count;
+---------------+
| @@error_ count |
+---------------+
|                0 |
+---------------+
1 row in set
```

5.2.3.3 Cluster and client character sets are inconsistent. Select Chinese field data export

When the character set on the cluster and client sides is inconsistent, if the table name, column name, or alias in the export select statement contains Chinese characters, it is necessary to set the client charset to gbk before execution. For example:

When the cluster character set is UTF8 and the client character set is gbk, when selecting Chinese fields for data export, it is necessary to connect to the database and specify the default character set of the cluster client as gbk. Otherwise, the export will report an error.

Tables and data used in the example:

```
gbase> set global gcluster_ extend_ ident=1;
Query OK, 0 rows affected (Elapsed: 00:00:00.01)

Gbase>create table t (day varchar (10), b varchar (10));
Query OK, 0 rows affected (Elapsed: 00:00:00.13)

Gbase>insert into t values ('china ',' country ');
```

Query OK, 1 row affected (Elapsed: 00:00:00.08)

Character set status:
The client editor character set is GBK
The cluster character set is
gbase> show variables like '%character_ set%';

```
+--------------------------+-------------+
| Variable_ name           | Value       |
+--------------------------+-------------+
| character_ set_ client   | utf8        |
| character_ set_ connection | utf8      |
| character_ set_ database | utf8        |
| character_ set_ filesystem | binary    |
| character_ set_ results  | utf8        |
| character_ set_ server   | utf8        |
| character_ set_ sort     | binary      |
| character_ set_ system   | utf8mb4     |
+--------------------------+-------------+
```

Direct export error, unable to recognize Chinese field names due to different character sets:

$ gccli
GBase client 9.5.2.39.126761. Copyright (c) 2004-2021, GBase. All Rights Reserved.
gbase> use testdb;

Query OK, 0 rows affected (Elapsed: 00:00:00.00)

Gbase>rmt: select days from tc into outfile '/home/gbase/2. dat' fields terminated by ',' WRITEMODE BY OverWRITES;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your GBase server version for the right syntax to use near '?? from tc into outfile '/home/gbase/2.dat' fields terminated by ',' WRITEMODE B' at line 1
Gbase>select days from tc into outfile '/home/gbase/2. dat' fields terminated by ',' WRITEMODE BY OverWRITES;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your GBase server version for the right syntax to use near '?? from tc into outfile '/home/gbase/2.dat' fields terminated by ',' WRITEMODE B' at line 1

When connecting to the database, specifying the client's default character set as gbk can successfully export the data
$ gccli --default_ character_ set=gbk
GBase client 9.5.2.39.126761. Copyright (c) 2004-2021, GBase. All Rights Reserved.

```
gbase> show variables like '%character_ set%';
+--------------------------+-----------+
| Variable_ name                    | Value         |
+--------------------------+-----------+
| character_ set_ client          | gbk             |
| character_ set_ connection | gbk             |
| character_ set_ database      | utf8            |
| character_ set_ filesystem | binary        |
| character_ set_ results        | gbk             |
| character_ set_ server          | utf8            |
| character_ set_ sort             | binary        |
| character_ set_ system          | utf8mb4      |
+--------------------------+-----------+
gbase> use testdb;
Query OK, 0 rows affected (Elapsed: 00:00:00.00)
Gbase>select days from tc into outfile '/home/gbase/2. dat' fields terminated by ',' WRITEMODE BY OverWRITES;
Query OK, 1 row affected (Elapsed: 00:00:00.07)
Gbase>rmt: select days from tc into outfile '/home/gbase/day_ 2.dat' fields terminated by ',' WRITEMODE BY OVERWRITES ;
Query OK, 1 row affected (Elapsed: 00:00:00.54)
```

5.2.4 Database Object Structure Export Tool gcdump

Function Description

GBase 8a MPP Cluster provides an export tool for database object structures, including table structures, stored procedures, and custom functions (excluding UDF and UDAF).

When using the gcdump tool to export the structure of a database object and generate the exported SQL script file, only the table structure is exported and the data is not included;

The Gcdump tool can only export one database object structure under the specified VC at a time.

Grammar format

```
gcdump [OPTIONS] database [tables]
gcdump [OPTIONS] --databases [OPTIONS] DB1 [DB2 DB3...]
gcdump [OPTIONS] --all-databases [OPTIONS]
```

OPTIONS parameter description

-A. -- all databases Export all user database structures

-u. -- user=name The database username used to connect to the database

-p. -- password [=name] The user password for connecting to the database

-P. -- port=# Port to connect to the database

-B. -- databases Export the specified database structure

-f. Ignore SQL errors during force export process

--Ignore table=database.table specifies the table not to be dumped. This parameter can only specify one table at a time. If multiple tables need to be ignored, use multiple parameters to specify.

-n. -- no create db does not output the database creation statement 'CREATE DATABASE IF NOT EXISTS db'_ name;' sentence

-t. -- no create info does not output table building statements

-q. Quick export results are not cached and are output directly

-Q. The table and column names output by quote names have reference symbols (`)

-r. -- result file=name Export the result to the specified file

-R. - Routines Export Stored Procedures and Functions

-W. -- fixed VC name=name specifies the name of the exported VC. Only one VC's database object can be exported at a time. If this parameter is not specified, it defaults to default VC

-X. - XML export file format is XML

-I. The colId export table structure contains TID and UID, the same as show full create table

Example

```
$ gcdump -A -W vc1 --ignore-table=testdb.t --ignore-table=testdb.abc
$ gcdump    -W vc1 -B testdb2
```

5.2.5 Use of Kafka Consumer Data Synchronization Function

5.2.5.1 Overview of kafka consumer

GBase 8a cluster supports kafka streaming data consumption and warehousing. It can achieve real-time synchronization of OLTP database data and program generated data into the 8a cluster.

Kafka is a third-party distributed publish subscribe message system with high throughput. Kafka supports a large amount of message data to enter the Kafka cluster system, persistence storage and queuing in the Kafka cluster, and waits for the message consumer to read the message.

GBase 8a cluster integrates the kafka consumer component, which supports users to

configure and manage consumer tasks in the 8a cluster through SQL, and synchronize messages from the kafka cluster to the 8a cluster database.

GBase 8a cluster supports running multiple kafka consumers, which can be connected to the same kafka server or different kafka servers.

The GBase 8a cluster also provides status monitoring for Kafka consumer data synchronization. Users can view the status of synchronization tasks (consumer tasks) by querying the system table.

The workflow of the Kafka consumer component in the 8a cluster is as follows:

1. The OLTP database publishes real-time changes in data information to the Kafka cluster through OGG or RTSync tools, or the data generated by user programs is published to the Kafka cluster through APIs.

The Kafka consumer component within the 8a cluster reads the published data information in real-time from the Kafka cluster, and converts this data information into database operations to be executed in the 8a cluster to achieve data synchronization.

The main function of kafka consumer is to read messages from the kafka cluster, parse the message content according to the message format, and convert the message content into database operations for execution in the 8a cluster. The Kafka consumer of the current 8a cluster supports parsing two types of database operations, namely Kafka transaction consumer and Kafka loader consumer.

- kafka transaction consumer

  Can parse insert, update (including full column update and non full column update), and delete operations. Other operations such as DDL and truncate are not supported.

  The transaction consumer uses a Kafka message as the basic unit of synchronization. A Kafka message can contain one or more database operations, and the operations contained in a message are either successfully executed or all fail in the 8a cluster.

  The message formats for transaction topics include JSON format and puredata format:

  JSON is text data that can be generated by user programs or by RTSync or OGG tools.

  Puredata is binary data, currently sourced from the RTSync tool.

  An example of a JSON format message is as follows:

```
{
"table":"BDTEST.TEST4",
"op_type":"I",
"op_ts":"2022-01-16 09:26:29.707674",
"current_ts":"2022-01-16T17:26:34.556001",
"pos":"00000000030000002194",
"after":{
        "A":4,
```

"B":40,
"C":"t4"
}
}

The converted database operation is as follows: insert into test4 values (4,40, 't4');

{
"table":"BDTEST.TEST4",
"op_type":"D",
"op_ts":"2022-01-16 09:36:44.703860",
"current_ts":"2022-01-16T17:36:49.047000",
"pos":"00000000030000003188",
"primary_keys":{"A"},
"before":{
"A":20
}
}

The database operation for conversion: delete from test4 where a=20;

{
"table":"BDTEST.TEST4",
"op_type":"U",
"op_ts":"2022-01-16 09:32:33.705303",
"current_ts":"2022-01-16T17:32:36.839000",
"pos":"00000000030000002612",
"primary_keys":{"A"},
"before":{
"A":2
}
"after":{
"A":20,
"B":200,
"C":"t20"

```
 }
}
```

The database operation for conversion is as follows: update test4 set a=20, b=200, c='t20 'where a=2;

Note: In the JSON message, "A" is recognized: "NULL" is a string with a NULL value for A, "A" is recognized: "null" is the empty value for A, and "A" is recognized: "" is the value for A.

- kafka loader consumer

Can parse load operations. The message content is the raw data that needs to be loaded, and the message format is the text format of the raw data itself.

There are the following differences between loader topics and transaction topics when creating consumers:

1. Because the message content of the loader consumer is raw data and does not contain library table information, it is necessary to specify the target table to load into when creating the consumer. The transaction consumer message contains target table information, so there is no need to specify it when creating a consumer.

2. The loader consumer supports consuming data from multiple partitions of the Kafka topic by default, and users can specify which partition to consume from. And transaction consumers only allow one partition in the corresponding kafka topic (because multi partition consumption cannot guarantee data order). Versions after 953.28 support multi partition consumption and require the parameter gcluster to be turned on_ kafka_ consumer_ support_ multi_ Partition=1, generally speaking, multi partition consumption cannot effectively improve performance, so it is generally not recommended to open it.

3. The consumer loader topic requires configuration of loading options (field spacing, line spacing, etc.), while the consumer transaction topic does not.

The operation process of kafka consumer is as follows:

1. Adjust kafka related parameters in the configuration files of each node of the cluster as required
2. Create consumer task
3. Start consumer task
4. The consumer reads the messages published by the task in real-time and stores them in the database
5. Through system table information_ schema.kafka_ consumer_ View the progress and status of Kafka transaction consumer consumption; Through system table information_ schema.kafka_ loader_ consumer_ View the progress and status of Kafka loader consumer consumption.

be careful

- The order of messages published to the Kafka cluster must be consistent with the order in which data changes occur. Reading messages by Kafka consumers will directly follow the order of messages in the Kafka cluster and execute them simultaneously.
- The performance of non full column updates in the message content synchronized by Kafka Consumer is slower than that of full column updates. Please use non full column updates reasonably during use.
- It is recommended that the kafka server use UTF8 (UTF8MB4) encoding, and messages published to the kafka server also use UTF8 (UTF8MB4) encoding. Because the format key character in the JSON message may be a byte of a certain man in GBK encoding, the Kafka server transcoding the JSON message may cause the JSON format to be invalid, causing the 8a cluster Kafka consumer to be unable to parse the read message.
- Kafka consumer supports task takeover, that is, if the consumer A to which Kafka consumer task 1 belongs shuts down abnormally due to software and hardware reasons, a new consumer B will be started on other good coordinator nodes to take over Kafka consumer task 1 and continue synchronization from the interruption of consumer A.
- After the kafka consumer task is started, it will continue to run until the user stops the task.
- In version 9.5.3.29, the table structure of the checkpoint table of the loader consumer was modified by adding a self increasing column to ensure the correct recording of the partition offsets corresponding to the last batch of data. If a user first creates a kafka loader consumer on an old version and has consumed data, they need to rebuild the loader consumer after upgrading to versions 9.5.3.29 and later, Alternatively, manually record the partition offset corresponding to the last data in the checkpoint table before upgrading, and then manually insert it into the checkpoint table after upgrading. Otherwise, there may be data duplication consumption issues.

5.2.5.2 **Parameter configuration**

To use Kafka Consumer, it is necessary to configure it in the following way. Please refer to the supplementary instructions for the configuration of parameters that can be changed.

1. Configure gcluster parameters

```
$GCLUSTER_ BASE/config/gbase_ 8a_ gcluster.cnf

_ gbase_ transaction_ Disable=1 (be sure not to use 0)

gcluster_ lock_ Level=0 (2 is not recommended)

_ gcluster_ insert_ cache_ buffer_ flag=1

gcluster_ assign_ kafka_ topic_ period=20
```

```
gcluster_ kafka_ max_ message_ size=1000000000

gcluster_ kafka_ batch_ commit_ dml_ count=500000

gcluster_ kafka_ local_ queue_ size=1010000

gcluster_ kafka_ consume_ batch=100

gcluster_ kafka_ user_ allowed_ max_ latency=15000
```

The above gcluster parameters are common for versions 95 and 86.

 **Description (changeable parameters)**

- gcluster_ assign_ kafka_ topic_ Period, the time period for automatically taking over the consumer, in seconds. For example, if node A goes down, the maximum waiting time is gcluster_ assign_ kafka_ topic_ After a period of seconds, the synchronization task taken over by node A will be taken over by other nodes. The minimum value is 20s, and the maximum value is 120s.

- gcluster_ kafka_ max_ message_ Size, the maximum length of a message obtained from the kafka topic, in bytes, with a maximum value of 1000000000 bytes. This value needs to be greater than or equal to the configuration of the kafka server (message. max. bytes), otherwise it may cause consumption issues. If there is a message in the kafka queue that exceeds gcluster in size_ kafka_ max_ message_ Size will cause consumption to get stuck.

- gcluster_ kafka_ batch_ commit_ dml_ Count, the number of dml operations submitted at once. Increasing it appropriately can significantly improve performance. However, if a topic involves many tables (hundreds of tables), it is recommended to reduce this parameter. The more tables there are, the smaller it should be. The purpose of reducing this parameter is to reduce the number of hit tables in one submission, and it needs to be treated specifically based on specific user scenarios, synchronization speed, and resource usage. After enabling new transactions in the future, the impact of a large number of tables on performance will be reduced, and the manual will be updated again. It should be noted that this parameter is an intention value, and the program may not strictly submit according to this parameter. For example, if a transaction contains a large number of DML operations, the program must ensure transaction integrity; For example, if the speed of retrieving and parsing messages from Kafka is slower than the speed of submitting data to a single machine, the program will also choose to submit first rather than waiting for gcluster to be met_ kafka_ batch_ commit_ dml_ The count parameter.

- gcluster_ kafka_ user_ allowed_ max_ Latency, how long messages are allowed to be cached in the GBase 8a MPP Cluster cluster layer, and must be submitted immediately after the timeout, in milliseconds. This parameter is related to gcluster_ kafka_ batch_ commit_ dml_ The function of count is similar, as it determines when to submit. Saving more data before submitting can help reduce disk usage. If users are less sensitive to data latency and more sensitive to disk usage, this parameter can be used to adjust. The typical value can generally be set between 50000 and 20000, and it should be noted that submitting the action itself also requires time.

- gcluster_ kafka_ local_ queue_ Size, the length of the queue for storing dml operations, recommended to be at least gcluster_ kafka_ batch_ commit_ dml_ More than double the count.

- gcluster_ kafka_ consume_ Batch, the number of Kafka messages read by the consumer at once. If the message size in the Kafka queue is small, it can be set to large. Conversely, setting it to small has little impact on performance, so it is generally not necessary to set it too large. It is recommended to set it to 10-1000.

- Switch parameters_ t_ kafka_ varchar_ auto_ Truncate: When a consumer consumes Kafka information and encounters a field with a length that exceeds the defined length in the database (only for varchar types), enabling automatic truncation and normal consumption and warehousing mode. The default value is 0; When the value is set to 1, it means that the consumer is asked to determine the length of the after content in the JSON message. If the length exceeds the column width of the target table, it will be automatically truncated according to the column width (character length), and only the varchar column will be processed.

- Control parameter: gcluster_ kafka_ message_ format_ Type
function: Set the format in which consumers parse kafka messages.
Value range: JSON, PUREDATA, AUTO_ DETECT
description:
Puredata corresponds to the protobuf message produced by rtsync;
AUTO_ DETECT (default) allows the consumer to detect the message format themselves. At this point, the consumer will first attempt to parse using puredata format, which is considered puredata format. Otherwise, it will be considered JSON format.
Note: After the consumer is started, this judgment is only made when parsing the first message, and the subsequent judgment result is directly used.

- Control whether a single consumer matches POS (preventing duplicate consumption) gcluster_ kafka_ ignore_ pos_ Field: Control whether a single consumer matches POS (to prevent duplicate consumption). Customers write data to kafka through multiple threads, and the data written to kafka cannot ensure POS order. When the original consumer consumes data, POS checks will be performed, resulting in missing out of order data when it is stored. Parameter gcluster_ kafka_ ignore_ pos_ Field, controls whether the consumer performs POS checks. POS check is enabled, and when the consumer consumes, the message before the consumed serial number will be discarded; The POS check is closed, and the consumer will store every message from Kafka, so the production end needs to ensure that there are no duplicate messages sent to Kafka.

gcluster_ kafka_ ignore_ pos_ field	Control whether the consumer performs POS checks The default value is 0, which means checking for duplicate messages; When the value is 1, do not check for duplicate messages	Applicable scenario: Consumer consumes only insert messages, and customers can ensure that there are no duplicate kafka messages in special scenarios

Configuration method:

Manually modify gclusterdb.kafka_ consumers

Update gclusterdb.kafka_ consumers set common_ options=’ gcluster_ kafka_ ignore_ pos_ field=1’ where `name`=’consumer_ 1’;

Restart consumer_ 1。

- Control the processing of delete operations:

gcluster_ kafka_ delete_ execute_ directly=100

When a batch of data is to be submitted, the program will count the number of delete operations for each table in the batch. If the number of delete operations exceeds gcluster_ kafka_ delete_ execute_ Directly, association deletion will be done through temporary tables, otherwise a delete statement will be issued directly.

 **be careful**

- The following parameters support independent configuration between consumers:

Number of dml operations submitted at once: gcluster_ kafka_ batch_ commit_ dml_ count

Delayed submission time: gcluster_ kafka_ user_ allowd_ max_ latency

Control whether a single consumer matches POS (to prevent duplicate consumption): gcluster_ kafka_ ignore_ pos_ field

Configuration method:

Manually modify gclusterdb.kafka_ consumers

Update gclusterdb.kafka_ consumers set common_ options=’ gcluster_ kafka_ batch_ commit_ dml_ count=10000,gcluster_ kafka_ user_ allowed_ max_ latency=1000,gcluster_ kafka_ ignore_ pos_ field=1’ where `name`=’consumer_ 1’;

Restart consumer_ 1。

2. Configure gnode parameters

86 version:

$GBASE_ BASE/config/gbase_ 8a_ gbase.cnf

_ gbase_ transaction_ Disable=1 (be sure not to use 0)

gbase_ tx_ log_ mode=ONLY_ SPECIFY_ USE (Be careful not to use USE, STANDARD_TRANS)

gbase_ buffer_ insert=1024M

gbase_ tx_ log_ flush_ time=5

Version 95:

$GBASE_ BASE/config/gbase_ 8a_ gbase.cnf

_ gbase_ transaction_ Disable=1 (be sure not to use 0)

gbase_ tx_ log_ mode=no_ use,no_ standard_ trans

_ gbase_ kafka_ transaction_ Mode=1 (must be turned on)

gbase_ buffer_ insert=1024M

gbase_ tx_ log_ flush_ time=5

explain

- gbase_ buffer_ Insert, size of insert buffer, with gcluster_ kafka_ batch_ commit_ dml_ Adjust the count setting. If the data volume is large and there are many consumer tasks, it is recommended to increase it. It is necessary to ensure that the single machine insert buffer is sufficient, otherwise it may cause exceptions.
- gbase_ tx_ log_ flush_ Time, the refresh rate of single machine memory data, in seconds. It is recommended to set it to 5 seconds.
- If the above parameters are not set properly, an error will be reported when executing the start kafka consumer command. Follow the error prompt to modify the parameter configuration.

be careful

- If gbase_tx_log_mode=USE, STANDARD was previously used on site_

  TRANS "This configuration (which was previously used in the development process and has been abandoned later), it is best to re export the data from the tables involved in the previous synchronized data after modifying the parameter" gbase_tx_log_mode=ONLY_SPECIFY-USE "(otherwise an error may be reported). If the data volume is large, the ignore parameter can also be configured_ gbase_ bsi_ check_ Disable=1 "Bypass this check error logic (the error generated by this situation will not cause any problems. The disadvantage of turning off the check is that it may not be able

to report an error in a timely manner if there are other bugs caus ing the error later on.)

- The configuration of each node should be consistent, and only modifying some nodes may result in unknown errors.

5.2.5.3 Consumer task operation command

5.2.5.3.1 premise

The following command is executed through gccli on any coordinator node.

5.2.5.3.2 Create consumer task

Grammar format

- transaction topic

```
CREATE KAFKA CONSUMER <consumer_ name> TRANSACTION TOPIC
<kafka_ topic_ name> BROKERS 'ip:port, ip:port,…';
```

- loader topic

```
CREATE KAFKA CONSUMER <consumer_ name> LOADER TOPIC <kafka_
topic_ name> BROKERS 'ip:port, ip:port,…' [PARTITIONS <partition list>]
DURATION <time in ms> INTO TABLE dbname.tbname <loader_ options>;
```

Table 5163 Parameter Description

Field Name	Explanation of Meaning
consumer name	The name of the consumption task. Unique, no duplicates allowed, maximum length 64 bytes;
kafka_ topic_ name	The name of the Kafka topic that needs to be consumed, with a maximum length of 64 bytes. Created by the user in advance on the kafka server. Note: When creating, the kafka topic name must already exist on the kafka server
Ip	Kafka broker's IP address.
Port	Kafka broker's port.
Partition list	Specify which/which partitions of the topic to consume data from, in the form of '0,1,2'. If all partitions are specified, all can

Field Name	Explanation of Meaning
	be used. The default is to consume data from all partitions of the topic. When creating a loader topic consumer, the partition must be a valid partition for the corresponding topic on the kafka server.
Time in ms	Elapsed time in milliseconds. The meaning is: The consumer task of the loader topic will tell the loader of gnode to continuously read data from the kafka server during the duration of the duration, and the data will be cached in memory. After reaching the duration, the data will be sent to the disk.
dbname.tbname	Database name. Table name, which means the target table where the consumer of the loader topic loads data.
Loader_ options	Inherited from the loading command options, the specific meaning and usage can refer to the cluster loading syntax instructions in Chapter 5.2.2, which can be set according to the format of the loaded data file. Specifically, it includes: Charset、data_ format、having lines、separator null_ value ‘’、fields terminated by ‘’、enclosed by ‘’、preserve blanks、length ‘’、table_ fields 、autofill、lines terminated by ‘’、max_ bad_ records 、datetime format 、date format 、timestamp format 、time format、parallel 、skip_ bad_ file 。

be careful

- Transaction topic type consumers do not allow two consumer tasks to use the same topic name+brokers combination.
- When creating a consumer, it is necessary that the topic in the Kafka cluster exists and the broker: port written can provide services normally.

Example

Create a consumer task named test1 from the topic_ 1. Consumption data:

```
Create kafka consumer test1 transaction topic topic_ 1
brokers ’10.10.10.10:9092,10.10.10.11:9092’;
```

5.2.5.3.3 Modify the loader topic consumer task

Only the partitions of the loader topic can be modified, and the loader topic consumer task can be modified when it is in the stop state.

Grammar format

```
ALTER KAFKA CONSUMER <consumer_ name> SET PARTITIONS
```

```
<partition_ offset>;
```

5.2.5.3.4 Delete consumer task

Function Description

Delete the consumer task.

be careful

- Before deleting, it is necessary to ensure that the consumer already exists and that the consumer task is in a stopped state, otherwise an error will be reported.

Grammar format

```
DROP KAFKA CONSUMER <consumer_ name>;
```

5.2.5.3.5 View consumer task properties

Grammar format

View all transaction consumer properties for data synchronization.

```
SHOW TRANSACTION CONSUMER;
```

View the loader consumer properties for all data synchronization.

```
SHOW LOADER CONSUMER;
```

View the properties of a single consumer, which are the parameters specified when creating the consumer task.

```
SHOW KAFKA CONSUMER <consumer_ name>;
```

Field Name	Explanation of Meaning
Name	consumer name。
Type	Consumer type
Topic	topic name
Brokers	Brokers List
Status	Current status of Consumer task Start is executing Stop stopped

Field Name	Explanation of Meaning
	Waiting start is currently starting
Db	Load the db name of the target table
Table	Load the table name of the target table
Partitions	Which partitions to load from
Duration	How long (ms) does the data land per consumption
Loader options	Load Format Options

5.2.5.3.6 Start consumer task

Function Description

Start the specified consumer task, which starts reading data from Kafka and synchronizing it into the warehouse.

To start, it is necessary to ensure that the consumer task already exists and is valid, and that the gcluster_ kafka_ consumer_ The enable value is 1.

Kafka Consumer supports breakpoint continuation and duplicate message screening.

After the consumer task stops and starts, breakpoint continuation will automatically resume synchronization from the position where the previous synchronization was completed.

Screening out duplicate messages means screening for duplicate messages and deleting them. This function can be configured through cluster parameters. If screening for duplicate messages is turned on, the performance will decrease compared to turning it off. If this function is turned off, a user program is required to ensure that the messages published to Kafka are not duplicate.

The consumer task scheduling mechanism ensures high availability and load balancing of consumer tasks. All consumer tasks that have already been started are allocated by the consumer task scheduling thread on which node they run. Therefore, consumer tasks may not necessarily run on the node where the command is executed. You can determine which node (IP address) they run on by checking the running status of consumer tasks.

During the operation of the consumer task, as long as the user does not execute the stop command to stop the consumer task, the consumer task will continue to run. Even if the cluster service stops completely and starts again, the consumer task will automatically continue to run.

Grammar format

Start the specified consumer task

```
START KAFKA CONSUMER <consumer_ name>;
```

Start all transaction topic consumers

```
START KAFKA TRANSACTION CONSUMER;
```

5.2.5.3.7 Stop consumer

Function Description

After execution, the specified consumer will be stopped.

Grammar format

Stop the specified consumer task

```
STOP KAFKA CONSUMER <consumer_ name>;
```

Stop all transaction topic consumers

```
STOP KAFKA TRANSACTION CONSUMER;
```

5.2.5.4 Status Query

Function Description

Query the synchronization status of all initiated consumer tasks.

Grammar format

Query the synchronization status of all initiated transaction topic consumer tasks.

```
SELECT * FROM information_ schema.kafka_ consumer_ status;
```

Query the synchronization status of all initiated loader topic consumer tasks.

```
SELECT * FROM information_ schema.kafka_ loader_ consumer_ status;
```

Table -5164 Parameter Description

Field Name	Explanation of Meaning
Consumer	consumer name。
IP	Which node (IP address) does the consumer task run on
Topic	kafka topic name。
Status	Current status of Consumer task Start is executing

Field Name	Explanation of Meaning
	Stop stopped Waiting start is currently starting
Min_ offset	The minimum offset in the current Kafka queue. If successfully connected to the Kafka server, this field is the actual value, otherwise it is 0.
Max_ offset	The maximum offset in the current Kafka queue. If successfully connected to the Kafka server, this field is the actual value, otherwise it is 0.
Cur_ offset	The offset of the message currently being obtained by GBase 8a MPP Cluster.
Process_ offset	GBase 8a MPP Cluster is executing offset for synchronized messages.
Partition_ offsets	The offset where the last successfully loaded landing data is located, including the offsets of each partition.
Commit_ offset	The offset of the corresponding message during the last commit operation of GBase 8a MPP Cluster.
Exception	The error description information when the last error occurred, including: unable to connect to the Kafka server, parsing JSON message error, and detailed error reasons, the target table does not exist, the target table column definition is incorrect (column name, number of columns, column order), an error occurred while processing data at the cluster layer, sending a single machine error message for execution, and submitting an error.

5.2.5.5 **Partial column update**

Usage scenario:

The update operation in Kafka Consumer JSON only includes the required columns, not all columns. In this case, the consumer will first query the values of all columns in the 8A library based on the primary key in JSON, then replace the column to be updated with the new value, and then perform a delete based on the primary key, and finally insert a row.

usage method:

1. Parameters to be configured:
gcluster_ kafka_ ignore_ pos_ Field=1, configure this parameter so that the consumer does not check POS.
_ t_ gcluster_ kafka_ ignore_ when_ update_ not_ When hit=1, when the query misses, the consumer ignores the current operation and does not synchronize it.

In this case, the OP in JSON_ TYPE="UN" instead of "U".

2. Add statistical function to calculate the delay time of each batch of data in the kafka consumer stage

Configuration parameters required:

gcluster_ kafka_ consumer_ latency_ time_ statistics = 1

After opening the parameters, the consumer will expand the number of columns in the checkpoint table by adding three columns: the time when this batch of data was received by the consumer, the time when the consumer completed submitting this batch of data, and the number of dml operations included in this batch of data.

3. Support escaping 0 in a string to ' 0'.

The occasional occurrence of a string containing 0 (note not "+'0') in the CMB project resulted in the final spelling of the consumer's SQL being truncated by this 0.

Need to configure parameter gcluster_ kafka_ consumer_ escape_ zero=1.

Non full column update optimization:

Operating steps:

1. The business side needs to ensure that the source side table structure is completely consistent with the table structure of 8a
2. Open non full column update control parameters
   gcluster_ kafka_ consumer_ support_ partial_ update
3. When confirming the update of a non-existent record, whether to report an error or discard the record to continue synchronizing data depends on the parameter_ t_

> An example of a message read by the consumer from Kafka during non full column updates is as follows:
> {“table”:”BDTEST.TEST4”,”op_type”:”UN”,”op_ts”:”2020-10-27 09:32:33.705303”,”current_ts”:” 2020-10-27 17:32:36.839000”,”pos”:”00000000030000002612”,”primary_keys”:{“A”},”before”:{“A”:2},”after”:{“A”:2,”B”:200}}
> The before field is the primary key information, and the after field is the field information that needs to be updated. The after field does not list all the updated values of the columns, which is equivalent to: update BDTEST. Test4 set A=2, B=200 where A=2;

   gcluster_ kafka_ ignore_ when_ update_ not_ Hit control, when the parameter value is 1, the record will be lost and synchronization will continue; When the parameter value is 0, an error is reported to stop synchronization.

Optimization implementation:

Non full column update optimization control parameter: gcluster_ kafka_ consumer_ support_ partial_ update 。 Turn on this parameter, the consumer supports non full column update function and optimizes performance; Turn off this parameter and the consumer will report an error when encountering a non full column update operation.

be careful:

After opening this parameter, it will increase the delay of all data operations (insert, delete, full column update, non full column update) for the consumer. The principle is as follows:

1. Non full column update is based on the update condition (primary key value), which first parses the entire record corresponding to the condition and extracts the full column data from 8a. Then, the entire record that will be extracted will be updated in full column, and the update statement will be converted into delete+insert execution.
2. In order to improve performance, non full column updates will be done in batches, i.e
   (1) Accumulated batch to commit_ Half of the batch
   (2) We have accumulated data for 2 seconds.
   The data collected in batches is stored in temporary tables, and then unified into the source table to extract the entire column of data for delete+insert. This way, for each target table, only one query action is executed, and the larger the batch, the lower the cost of a single entry.
3. The two stages of parsing and warehousing are serialized. After the previous batch of data has been submitted for warehousing, the next batch of data is parsed. This serialization of queries and submissions optimizes the use of hard drives.

Note:

1. Consumer's support limit for non full column update operations: When this row of data is not in the library, it can cause data loss
2. The performance improvement of non full column updates after optimization is still quite significant, but due to the current implementation, it is still difficult for non full column updates to fully achieve the efficiency of full column updates.

5.2.5.6 **Consumer loose mode**

Usage scenario:

The update operation in Kafka Consumer JSON only includes the required columns, not all columns. In this case, the consumer will first query the values of all columns in the 8A library based on the primary key in JSON, then replace the column to be updated with the new value, and then perform a delete based on the primary key, and finally insert a row.

Consumer's support for data operations in relaxed mode:

1. Set the loose mode. After both parameters are set to 0, turn on the loose mode:
   _ t_ gcluster_ kafka_ consumer_ force_ compare_ Set field to 0
   The default value of this parameter is 1, which means it is open. The consumer will strictly compare the field name and order for each JSON message to ensure that there is no DDL in the source database to the greatest extent possible.
   _ t_ gcluster_ kafka_ consumer_ compare_ field_ only_ Once set to 0
   The default value of this parameter is 0, and setting it to 1 means it is open. When opened, it means that the consumer will only strictly compare the field names and order when it first encounters a JSON message from the t table. After the comparison is passed, it will no longer be compared and will be parsed according to the earliest JSON. In this

case, some optimization methods will be used. When it is possible to ensure that the source database will not perform DDL operations, it can be considered to open it, but it is generally not recommended to open it.

2. The consumer's requirements for field names and order in relaxed mode are as follows:
   Condition 1: Field names must be capitalized;
   Condition 2: Allow any JSON message about the t table, as long as each field belongs to the target table (allowing field names and order changes). For example:
   Target Table: t (A not null, B not null, C default 2, D default null)
   JSON_ 1: insert into t (A, C, B) OK
   JSON_ 2: insert into t (A, B, C) OK
   JSON_ 3: insert into t (A, B) OK
   JSON_ 4: Insert into t (B, C) parsing is OK, but an error is reported when entering the warehouse.
   The table has specified that column A cannot be empty, and users need to ensure it themselves. JSON_ 4 is equivalent to insert into t (A, B, C) values (NULL, xx, xx)
   Condition 3: For the missing fields in Condition 2, the default attribute should be specified when creating the table, otherwise there may be errors during the warehousing stage, which is guaranteed by the user themselves.
3. Consumer's DDL support remains unchanged in loose mode:
   Consumer does not support synchronization of DDLs other than truncate

 **explain**

Explanation of support in previous versions in non loose mode

1. Field name must be capitalized

2. The first JSON message about table t is allowed to not provide all fields, and the order of fields is also allowed to be inconsistent with 8a. However, subsequent messages must also remain the same, otherwise an error will be reported.

3. The missing fields in condition 2 should have the default attribute specified when creating the table, otherwise there may be errors during the warehousing stage. This is guaranteed by the user themselves.

5.3 Database performance optimization

For databases, efficiency is one of the most important indicators. To improve the database performance, we should do a good job in the following aspects: database design, SQL statement optimization, database parameter configuration, appropriate data resources and operating system.

5.3.1 Parameter configuration

be careful

For a single SQL, there are scenarios where configuring a certain parameter may significantly improve performance. Therefore, it is necessary to modify the configuration parameters reasonably based on the actual business scenario. At the same time, due to the global nature of configuration parameter adjustments, it is necessary to be cautious in making global parameter adjustments when it is not possible to ensure good results for the overall business.

5.3.1.1 Operating System Parameter Configuration

Example of initializing operating system related parameters:

1. SWAP

   It is recommended to place the SWAP file and data file on different disks.

   When the physical memory is greater than 128GB, it is recommended to set it to half of the physical memory, not less than 64GB. When the physical memory is less than 128GB but greater than 64GB, it is set to twice the physical memory. When the physical memory is less than 64GB, it should not exceed twice the physical memory.

2. VFS_ CACHE_ Pressure

   Trend of data writing to disk: recommended value is 100-1024.

   When the average memory per core is small, the value can be adjusted to be higher.

3. virtual memory

   Default value: limited limits memory usage.

```
set virtual memory=unlimited
```

4. Disk scheduling strategy

   The disk scheduling strategy algorithm cannot be 'cfq' (but drives that include swap space, log files, and Linux system files can still use this method). It is recommended to use 'deadlock' for disks or 'loop' for SSDs, such as executing the following command:

```
echo deadline > /sys/block/sda/queue/scheduler
```

   Settings need to be made based on the actual device where the installation directory is located.

5. Transparent page management

   Transparent page management cannot be enabled and must be closed. Execute the following command:

```
echo never > /sys/kernel/mm/transparent_ hugepage/enabled
```

6. Kernel parameter settings

   Set the threshold for system memory recycling, control the idle memory of the system, and configure it in the/etc/sysctl. conf file.

```
vm.vfs_ cache_ pressure =    1024;
```

   vm.min_ free_ Kbytes=1/10 of the size of physical memory, please note that the unit of this parameter is kbytes.

7. Memory and virtual memory settings

   The limits of memory and virtual memory need to be set to unlimited.

```
max memory size             (kbytes, -m)          unlimited
```

8. CPU overclocking

   Turn off CPU overclocking, which can be set in the BIOS.

9. I/O 调度方式

   The deadline scheduling algorithm is recommended for mechanical hard drives, and the noop scheduling algorithm is recommended for solid-state hard drives.

10. FD related parameters

```
HARDFDLIMIT="65536"
SOFTFDLIMIT="65536"
```

5.3.1.2 Database system parameter configuration

System parameters are read from the configuration file during server initialization, and parameters with no set values use the system default values.

- How to set system parameters:
  - Set in the configuration file that the service needs to be restarted to take effect.
  - Set after connecting to the database.

    View the value of system parameters, show variables like 'parameter name';

    Set [global] system variable=value

    It takes effect in the current session or globally, and will no longer take effect after the service is restarted.

be careful

System parameters are optimized for a specific scenario, and most optimizations are enabled by default. Only when certain SQL statements perform poorly, can we consider adjusting system parameters.

5.3.1.2.1 GNode parameter optimization

5.3.1.2.1.1 Memory parameters of GNode

1. **Heap parameter (global)**

   gbase_ heap_ The main design purpose of data is to cache data (DC) and allocate the most memory.

   gbase_ heap_ Large is used to manage infrequently requested and released memory.

   gbase_ heap_ Temp is used to allocate more trivial and small temporary memory, and is less commonly used.

   gbase_ memory_ pct_ Target sets the available proportion of memory, with a default of 0.8.

- Parameter lower limit:

```
gbase_ heap_ data   >= 512MB
gbase_ heap_ large   >= 256MB
gbase_ heap_ temp   >= 256MB
```

- Parameter upper limit:

```
(gbase_heap_data + gbase_heap_large + gbase_heap_temp ) <= total memory * gbase_ memory_ pct_ target
```

be careful
Default:
total memory = physical memory
_ gbase_ memory_ use_ When swap is set to 1:
total memory = physical memory + swap

2. **Operator buffer (session)**

   The operator buffers are all at the session level, that is, if gbase is set_ buffer_ Result=1G, and the number of concurrency is 30, then during the execution of concurrency, the total gbase occupied by 30 concurrency is_ buffer_ The result is 1G * 30=30G, and it is without calculating other operator buffers.

   So if a certain operator buffer is set to a large size in a high concurrency environment, it is highly likely that there will be insufficient memory to allocate.

- The commonly used buffers are as follows:
  - gbase_ buffer_ Distgrby: used to save the intermediate results of the distinct operation;
  - gbase_ buffer_ Hgrby: used to save the intermediate results of the hash group by operation;

- gbase_ buffer_ Hj: used to save the intermediate results of the hash join operation;
- gbase_ buffer_ Insert: used to save the intermediate results of insert values;
- gbase_ buffer_ Result: used to store intermediate results of materialization;
- gbase_ buffer_ Rowset: used to save the intermediate result set of join calculations;
- gbase_ buffer_ Sj: used to save the intermediate results of a sort merge join. When the join condition is a>=b or a<=b, a sort merge join may be used;
- gbase_ buffer_ Sort: used to save the intermediate results of a sort operation.

- The setting principle of operator buffer:
  In general (non high concurrency scenarios), based on the system memory size, the operator buffer can be set as follows:
  - gbase_ buffer_ Hgrby and gbase_ buffer_ Hj maximum not exceeding 4G;
  - gbase_ buffer_ The maximum result should not exceed 2G;
  - gbase_ buffer_ Rowset maximum not exceeding 1G;
  - Other operators can be estimated using the system.
  - If in a high concurrency scenario, there is no need to set an excessively large operator buffer, which is generally based on the system's automatic evaluation. But if the number of concurrency is too large, it is not ruled out that it is necessary to manually set the operator buffer to a smaller size. The number of concurrent operations multiplied by the total operator buffer size does not exceed gbase_ heap_ Large is recommended, but the maximum size cannot exceed the total system memory size.
- Other scenarios for modifying the operator buffer:
  If a certain SQL statement is executed too slowly due to a certain operator (bottleneck point can be referred to as a single machine trace), the corresponding operator buffer can be appropriately increased. For example, according to the trace, if the join is found to be slow, the gbase can be appropriately increased_ buffer_ The value of hj. However, it should be noted that adjusting this value must not affect the execution of other SQL statements.

5.3.1.2.1.2Concurrency control parameters for Gnode

1. **gbase_ parallel_ execution**

- Used to set whether to enable parallel switches.
  0 off, default to off.
  1 On.

- Applicable scenario:

  When low CPU utilization is found, parallelism can be turned on.

  The principle of parallelism is to divide data into multiple blocks, process them in parallel, and finally merge the result set

  Suitable for complex SQL scenarios with low concurrency.

2. **gbase_ parallel_ max_ thread_ in_ pool**

- It is used to configure the maximum number of threads in the parallel executor thread pool, which is twice the number of system CPU cores by default.

  Value range: 0-4096, default to the number of CPU cores in the system, and should not exceed 4 times the number of CPU cores.

 **explain**

The threads in the thread pool are created when the database service starts, leased from the thread pool when SQL is executed, and returned after use, which can effectively avoid the cost of frequent thread creation and destruction.

3. **gbase_ parallel_ degree**

- Control the maximum parallelism of each SQL

  Value range: 0~gbase_ parallel_ max_ thread_ in_ Pool, the maximum value range does not exceed the maximum number of available threads in the thread pool.

  1 indicates that parallelism is not started, meaning single threaded execution.

  0 means the default parallelism is thread pool gbase_ parallel_ max_ thread_ in_ 1/2 of the pool (if gbase_parallel_max_thread_inpool is odd, only the integer part will be taken).

5.3.1.2.2 GCluster parameter optimization

1. **Query optimization parameters**

- gcluster_ hash_ redistribute_ groupby_ optimize

  Cluster use hash redistribute groupby mode, default is 1(0 - OFF, 1 - ON)

  If this option is enabled, before performing group by operations, the temporary results will be redistributed to each operation node using a hash algorithm, and then grouped by each node. Due to the fact that the data has already been hashed before being divided into various nodes, the resulting results can be directly summarized to obtain the final result, and there is no longer a need for the summarizing node to group again.

- gcluster_ hash_ redistribute_ join_ optimize

This parameter is used to control whether to enable the JOIN mode of hash redistribution, with a default of 2 (0- pull replication table, 1- dynamic hash, 2- automatic evaluation)

If this option is enabled, when performing an equal join operation on two distribution tables, the data of one table will be dynamically hashed based on the value of the join condition column. Then use the temporary table dynamically hashed on each operation node and another table for join operation. In this way, the calculation results of each node can be directly summarized to obtain the final result. This strategy does not pull one of the distribution tables into a replication table on all computing nodes, and each computing node only needs to receive a portion of the data from this table. If the parameter is set to 2, use hash redistribution JOIN when the difference in data volume between the two tables does not exceed 20%; Otherwise, it will not be used.

- gcluster_ special_ correlated_ optimize

  This parameter is used to control whether to enable related sub query hash redistribution optimization. If a parent-child query is a related child query relationship and there is an equivalent JOIN relationship, the parent-child query will be hashed and redistributed according to the JOIN column before execution.

  - Parameter=0 off;
  - Parameter=1 enabled;
  - The default value of this parameter is 1.

  The use of equivalent hash redistribution related sub query function requires coordination with parameter gcluster_ crossjoin_ use_ hash_ The principles for using distribution together are as follows:

  - When gcluster_ special_ correlated_ When optimize=0, regardless of setting the parameter gcluster_ crossjoin_ use_ hash_ The value of distribution will not enable this optimization.
  - When gcluster_ special_ correlated_ Optimize=1, while gcluster_ crossjoin_ use_ hash_ When distribution=0, this optimization will not be enabled.
  - When gcluster_ special_ correlated_ Optimize=1, and gcluster_ crossjoin_ use_ hash_ This optimization will only be enabled when distribution=1.

  An example is as follows:

```
SELECT COUNT(*) FROM x1 WHERE EXISTS (SELECT 1 FROM x2 WHERE
x1.id2 = x2.id2);
```

  Equivalent hash related sub query optimization, dynamically redistributing x1 and x2 respectively.

- gcluster_ crossjoin_ use_ hash_ distribution

  This parameter is used to set whether to still force the hash redistribute JOIN when neither side of the JOIN is a hash column.

  - Parameter=0: indicates off, that is, when neither side of the JOIN is a hash column, the hash redistribute JOIN function is not used;
  - Parameter=1: indicates on, that is, when neither side of the JOIN is a hash column, the hash redistribute JOIN function is used;
  - The default value of this parameter is 1.
- gcluster_ empty_ result_ set_ optimize

  This parameter is used to set whether to enable the empty result set optimization function. The default is 0 (0-OFF, 1-ON). Empty result set optimization. If the optimization stage can determine that the result set is empty, it will be returned directly without the need for the executor to execute.
- gcluster_ single_ hash_ node_ optimize

  Optimization of single table hash conditions. When a single table contains equivalent conditions for hash columns, hash optimization is performed, and SQL statements are only sent to a single node. The default is 1 (0- OFF, 1- ON).
- gcluster_ hash_ join_ complex_ optimize

  The optimization of sub queries and parent queries that satisfy hash relationships can be executed as a whole. This parameter is used to control whether to execute as a whole when optimizing sub queries and parent queries that meet the hash relationship.

  - When parameter=0 is disabled, optimization that satisfies the hash relationship should not be executed as a whole or part;
  - When parameter=1 is enabled, optimization that satisfies the hash relationship is executed as a whole part.
  - The default value of this parameter is 1.
- gcluster_ union_ optimize

  This parameter is used to set whether to use union optimization. When using union optimization, it is best to send union statements to nodes for execution, avoiding pulling all tables that require union into replicated tables. By utilizing the deduplication feature of the union's result set, the union is directly sent to the lower level for execution. In some cases, this can greatly reduce the size of the intermediate result set of the summary node.

  - Parameter=0: Do not use union optimization;
  - Parameter=1: Use union optimization.

- Default value is 1

- gcluster_ starschema_ join_ estimate_ optimize

  Set the method for evaluating the join results of two tables, 0: evaluate by multiplying the number of rows in the two tables; 1: Evaluate based on the number of rows in the large table of the two tables, with a default value of 1.

- gcluster_ delayed_ group_ by_ optimize

  Set whether group by is distributed to gnode for execution. When the result set does not significantly decrease after group by is distributed to gnode for execution, please set this parameter to 1, (0- OFF, 1- ON).

- gcluster_ count_ optimize

  When setting count (*) on a single table, no intermediate result table is generated, and gcluster directly calculates the result value.

  - 0: Execute according to the method of generating intermediate tables;
  - 1: Gcluster directly calculates the count value.

  Optimization point: The original strategy for count (*) was to first create a temporary summary table on the gnode on the initiating node, summarize the execution results of each node onto the temporary summary table, and then perform a sum summary on the temporary summary table. After the summary is completed, the temporary summary table is deleted. When there is high concurrency, frequent creation and deletion of temporary summary tables will reduce performance. Now, it is modified to not create temporary summary tables, Collect the execution results of each node on gcluster and calculate them directly, so there is no need to create or delete temporary summary tables, thereby improving performance.

2. Concurrent parameters

- gcluster_ serial_ exec_ query

  Optimization point: Because gnode does not have automatic resource management capabilities, when high concurrency occurs, gnode's execution performance decreases due to resource contention among various concurrent SQL statements. Therefore, gcluster can be used to control the number of SQL statements sent to gnode to achieve indirect control over gnode's use of resources, so that gnode's various concurrent SQL statements do not compete for resources, thereby improving performance. This can be achieved by setting gcluster in the gcluster configuration file_ serial_ exec_ Query=Batch submission count (single node CPU cores) to control the number of SQL submissions to gnode.

  This parameter defaults to 0 (unlimited).

- gcluster_ max_ conn_ in_ pool

Optimization point: when there is no thread pool, the concurrent number of gcluster accesses to gnodes will not be controlled. Each gnode accessed by gcluster will start a new thread. When there is high concurrency, a large number of threads will consume system resources and increase the pressure on gnodes. When thread pool is adopted, concurrent requests compete for threads in the same thread pool. Therefore, the concurrent number of gcluster accesses to gnodes can be strictly controlled through the maximum number of threads, This reduces the pressure on gnode and reduces the consumption of system resources by threads.

This parameter defaults to 300.

- gcluster_ use_ conn_ pool

  Optimization point: when there is no Connection pool, the number of connections that gcluster accesses to gnodes will not be controlled. Each gnode that gcluster accesses will start a new connection, which will increase the connection time to gnodes. In addition, a large number of connections will consume system resources when there is high concurrency. When Connection pool is adopted, concurrent requests compete for connections in the same Connection pool. Therefore, The maximum number of connections can strictly control the concurrent number of gcluster accessing gnode, thereby reducing the pressure on gnode and reducing the consumption of system resources by connections.

  (0-OFF, 1-ON)

- gcluster_ insertselect_ use_ values_ optimize

  Optimization point: When multiple inserts into t1 select * from t are highly concurrent, they can only be executed serially on gnode, affecting execution efficiency. However, if the method of insert into t1 values () is followed, concurrent execution is allowed.

  Use case:

```
INSERT into TB_ SVC_ SUBS_ HIST_ TMP1
SELECT *
FROM TB_ SVC_ SUBS_ HIST
WHERE
MSISDN=MSISDN='xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'
```

  Applicable scenario: Insert select has high concurrency and the result set of the select is not very large.

  (0-OFF, 1-ON)

- gcluster_ single_ hash_ node_ optimize

  Used for optimizing single table equivalent hash query conditions. When a single

table contains equivalent conditions for hash columns, hash optimization is performed, and SQL statements are only sent to a single node.

Default value is 1: On

Example:

```
select * from bas.clcinfdta where 1=1 and clt_ nbr='7319022720' limit 30 offset 0
```

Unable to find records, remove 1=1 to correctly query

Can set gcluster_ single_ hash_ node_ Optimize=0, temporarily closed

- _ gbase_ enable_ hashtree

  The default value is 1 (enabled), which is inefficient when used for joining due to repeated connection values;

  When the value is 0, the join does not enable hashtree, but uses linked list structural join.

- gcluster_ ddl_ parallel_ execute

  This parameter is used to control DDL parallel or serial execution.

  - Parameter=0: serial execution;
  - Parameter=1: Parallel execution
  - The default value for this parameter is 0

5.3.1.2.3 Optimization of OLAP windowing function parameters

In the window opening function of gbase, the performance of sum and avg decreases with the size of the window. You can adjust the parameter gbase_ buffer_ Rowset value to improve performance. gbase_ buffer_ Rowset represents the maximum memory limit for saving intermediate result sets.

- Execution principle:

When performing olap calculations, the first step is to use gbase_ buffer_ The rowset parameter allocates memory space for storing intermediate result sets for the calculation process, and sets the gbase_ buffer_ Rowset is divided into n blocks, where n is determined by the maximum number of threads and parallel_ The degree determines that these buffers can be used during the calculation process. Then, since only the OLAP column information is saved during the calculation process, the buffer is divided by the column width of the OLAP column to obtain the number of rows that can be represented by a buffer. At this time, if the number of rows in a partition exceeds the upper limit of the number of rows saved by a buffer, the excess is saved in the next buffer. When sliding the calculation line by line in the window, if a partition is saved across buffers, the data will be accessed across buffers during the sliding process, This will result in

frequent swapping in and out, with the maximum number of times the window can be reached. Therefore, if the window is too large and there are large partitions, it will cause frequent swapping in and out, resulting in performance loss.

- Parameter settings:

Evaluate the data volume of the largest partition in the data table in advance and adjust the gbase according to the following formula_ buffer_ The value of rowset should increase the memory of the buffer and try to save the maximum partition within one buffer to minimize the number of cross buffer accesses.

max_ partition_ Size * column width (column calculated using the olap function, such as sum (a), if column a is of type double, the column width is 8) to obtain a buffer value. Then consider gbase_ parallel_ The degree parameter. If the degree value is 0, multiply the value of this buffer by the maximum number of threads gbase_ parallel_ max_ thread_ in_ After taking half of the pool to the nth power of 2 (such as 1024512), a more suitable rowset value is obtained.

If gbase_ parallel_ If the degree parameter is not 0, multiply the value of buffer by the maximum number of threads gbase_ parallel_ max_ thread_ in_ Pool, take the result up to the nth power of 2, and obtain a more suitable rowset value.

Finally, reset the gbase_ buffer_ rowset。

- Summary:

if(dregree == 0)

max_ partition_ Size * column width * gbase_ parallel_ max_ thread_ in_ pool/2 = rowset;

else

max_ partition_ Size * column width * gbase_ parallel_ max_ thread_ in_ pool = rowset;

For example:

```
create table sales(
sales_ employee varchar(50) not null,
fiscal_ year varchar(50) not null,
sale decimal(14,2) not null,
primary key(sales_employee,fiscal_year)
);
select
    fiscal_ year,
    sales_ employee,
```

```
    sale,
    avg(sale) over (partition by sales_employee order by fiscal_year
  ROWS BETWEEN 10000 PRECEDING AND 10000 FOLLOWING) total_ sales
from   sales;
```

Load data of 40 million; Set parameter gbase_ buffer_ Rowset=3073741824, with relatively stable execution performance and no performance fluctuations due to window size changes.

5.3.2 performance optimization

5.3.2.1 resource management

The challenges faced by GBase 8a MPP Cluster cluster:

1. When system resources are not controlled, all SQL executions will seize resources, which can cause system instability;
2. The system resources are heavily occupied by a low priority SQL, resulting in the inability to complete emergency SQL on time;
3. Complex SQL is often executed in multiple steps in a cluster, and in concurrent situations, tasks of the same SQL are limited by resources and cannot be guaranteed to be completed synchronously across all nodes.

So the problem that resource management needs to solve is:

1. System resources can be allocated and used according to policies;
2. The execution of tasks should have priority management;
3. Complex (multi-step) tasks in a cluster need to have a unified management strategy (including resource allocation, priority, execution order, etc.).

5.3.2.1.1 Cluster related commands

1. User priority setting

- Grammar:

```
grant usage on *.* to user_ name with task_ priority priority_ value
```

priority_ The value range is 0, 1, 2, and 3 corresponding to the minimum priority, low priority, medium priority, and high priority. The default is medium priority 2.

- Permission requirements:

  Users with grant privileges, recommended user: root

- Example:

```
gbase> create user uer1 ;
Query OK, 0 rows affected

gbase> grant usage on *.* to    uer1    with task_ priority    1;
Query OK, 0 rows affected

gbase> select task_ priority from user where user='user1';
+---------------+
| task_ priority |
+---------------+
|                      1 |
+---------------+
1 row in set
```

2. User resource group settings

- Grammar:

```
grant usage on *.* to user_ name with resource_ group group_ value
```

group_ The value range is 0-15, and the 0 group is the default group.

- Permission requirements:

For users with grant privileges, the recommended user is root.

- Example:

```
gbase> create user user0 identified by 'user0';
Query OK, 0 rows affected

gbase> grant usage on *.* to user0 with resource_ group 0;
Query OK, 0 rows affected

gbase> select resource_ group from user where user='user0';
+----------------+
| resource_ group |
+----------------+
|                       0 |
+----------------+
1 row in set
```

3. Query parallelism setting

- Grammar:

```
grant usage on *.* to user_ name with max_ cpus_ used max_ cpus_ used_ value
```

max_ cpus_ used_ The value range is an integer greater than 0, which determines the parallelism of the query. The recommended value is the number of available CPUs in the user's resource group.

- Permission requirements:

For users with grant privileges, the recommended user is root.

- Example:

```
gbase> use gbase;
Query OK, 0 rows affected

gbase> create user user1;
Query OK, 0 rows affected

gbase> grant usage on *.* to    user1    with max_ cpus_ used    4;
Query OK, 0 rows affected

gbase> select max_ cpus from user where user='user1';
+----------+
| max_ cpus |
+----------+
|              4 |
+----------+
1 row in set
```

4. User priority and task scheduling weight setting

- grammar

```
Set    gcluster    global    gbase_ high_ priority_ weight = weight_ value（80-100）
Set    gcluster    global    gbase_ mid_ priority_ weight = weight_ value （60-80）
Set    gcluster    global    gbase_ low_ priority_ weight = weight_ value （40-60）
Set    gcluster    global    gbase_ min_ priority_ weight = weight_ value （20-40）
```

- weight_ The value values are divided into high, medium, low, and minimum, and the specific range is as follows:

Height: 80-100

Medium: 60-80

Low: 40-60

Min: 20-40

- Permission requirements

  Users with set permission

- Example

```
gbase> Set  gcluster  global  gbase_ min_ priority_ weight = 20;
Query OK, 0 rows affected, 32 warnings (Elapsed: 00:00:00.01)

gbase> show variables like '%gbase_ min_ priority_ weight%';
+--------------------------+-------+
| Variable_ name                    | Value |
+--------------------------+-------+
| gbase_ min_ priority_ weight | 20       |
+--------------------------+-------+
1 row in set (Elapsed: 00:00:00.00)
```

explain

The counterweight value of the above statement will not be persistent, that is, it will be lost after the restart of gnode.

If persistence is required, execute the following statement before executing the statement:

```
set  gbase_ global_ variable_ persistent = 1
```

After executing this statement, execute:

```
set  gbase_ global_ variable_ persistent = 0
```

The counterweight parameters can control the CPU. shares and BLKIO. weight parameters.

Table 5-165 for specific reference

parameter	CGroup minimum value	CGroup maximum value	CGroup default value	colony minimum value	colony Maximum value	Calculation of cluster proportion
cpu.shares	one	nothing	one thousand and twenty-four	one	two thousand five hundred and sixty	(2560 * weight) / 100
Blkio.weight	one	one thousand	three hundred	one	one thousand	(1000 * weight) / 100

5. Display priority status

1) grammar

Show priorities [where conditions]

2) display

node_ Name: The name of the cluster node.

Group: Resource group number.

Priority: Priority number.

priority_ Weight: Priority counterweight.

Status: Priority on status ON/OFF.

Description: Priority control parameter description.

3) Permission requirements

Users with show permission.

4) Example

- The cgconfig.conf file needs to be configured. Set resource groups 0 and 1 in the configuration file.
- Start the cgconfig service.
- Restart gcware:

$gcluster_ services gcware restart

After completing the above configuration operations, execute the command in the following example, and the priority of the control groups 0 and 1 will be turned on.

- Example 1: View the priority status of all nodes in the cluster.

```
gbase> show priorities;
+-----------+-------+----------+-----------------+--------+-------------+
| node_ name | group | priority | priority-weight | status | description|
+-----------+-------+----------+-----------------+-------+-------------+
| node1      |     0|        0 |              20 | ON     |.....        |
| node1      |     0|        1 |              40 | ON     |.....        |
| node1      |     0|        2 |              60 | ON     |.....        |
| node1      |     0|        3 |              80 | ON     |.....        |
| node1      |     1|        0 |              20 | ON     |.....        |
| node1      |     1|        1 |              40 | ON     |.....        |
| node1      |     1|        2 |              60 | ON     |.....        |
```

```
| node1      |      1|      3 |                 80 | ON       |.....          |
| node1      |      2|      0 |                 20 | OFF      |.....          |
| node1      |      2|      1 |                 40 | OFF      |.....          |
| node1      |      2|      2 |                 60 | OFF      |.....          |
| node1      |      2|      3 |                 80 | OFF      |.....          |
.....
| node2      |     15|      0 |                 20 | OFF      |.....          |
| node2      |     15|      1 |                 40 | OFF      |.....          |
| node2      |     15|      2 |                 60 | OFF      |.....          |
| node2      |     15|      3 |                 80 | OFF    |.....          |
+-----------+-------+----------+-----------------+-------+-------------+
128 rows in set
```

- Example 2: Viewing the priority status information of node1 node.

```
gbase> show    priorities    where node_ name = 'node1';
+-----------+-------+----------+----------------+--------+-------------+
| node_ name | group | priority | priority-weight | status | description|
+-----------+-------+----------+-----------------+-------+-------------+
| node1      |      0|      0 |                 20 | ON       |.....          |
| node1      |      0|      1 |                 40 | ON       |.....          |
| node1      |      0|      2 |                 60 | ON       |.....          |
| node1      |      0|      3 |                 80 | ON       |.....          |
| node1      |      1|      0 |                 20 | ON       |.....          |
| node1      |      1|      1 |                 40 | ON       |.....          |
| node1      |      1|      2 |                 60 | ON       |.....          |
| node1      |      1|      3 |                 80 | ON       |.....          |
| node1      |      2|      0 |                 20 | OFF      |.....          |
| node1      |      2|      1 |                 40 | OFF      |.....          |
| node1      |      2|      2 |                 60 | OFF      |.....          |
| node1      |      2|      3 |                 80 | OFF      |.....          |
.....
| node1      |     15|      0 |                 20 | OFF      |.....          |
| node1      |     15|      1 |                 40 | OFF      |.....          |
| node1      |     15|      2 |                 60 | OFF      |.....          |
| node1      |     15|      3 |                 80 | OFF      |.....          |
+-----------+-------+----------+-----------------+-------+-------------+
```

```
64 rows in set
```

- Example 3: Viewing priority information with a status of ON.

```
gbase> show priorities where status ='ON';
+-----------+-------+----------+----------------+--------+-------------+
| node_ name | group | priority | priority-weight | status | description|
+-----------+-------+----------+----------------+-------+-------------+
| node1      |     0|        0 |              20 | ON     |.....        |
| node1      |     0|        1 |              40 | ON     |.....        |
| node1      |     0|        2 |              60 | ON     |.....        |
| node1      |     0|        3 |              80 | ON     |.....        |
| node1      |     1|        0 |              20 | ON     |.....        |
| node1      |     1|        1 |              40 | ON     |.....        |
| node1      |     1|        2 |              60 | ON     |.....        |
| node1      |     1|        3 |              80 | ON     |.....        |
+-----------+-------+----------+----------------+-------+-------------+
8 rows in set
```

- Example 4: Turn off the service cgconfig stop of node 1 cgroup configuration service.

```
# service cgconfig stop
Stopping cgconfig service: [    OK    ]
# su - gbase
$ gcluster_ services all restart
Stopping GCMonit success!
Signaling GCRECOVER (gcrecover) to terminate: [    OK    ]
Waiting for gcrecover services to unload:... [    OK    ]
Signaling GCSYNC (gc_sync_server) to terminate: [    OK    ]
[    OK    ]for gc_ sync_ server services to unload:[    OK    ]
Signaling GCLUSTERD    to terminate: [    OK    ]
.[    OK    ]or gclusterd services to unload:... [    OK    ]
Signaling GBASED    to terminate: [    OK    ]
.[    OK    ]or gbased services to unload:[    OK    ]
Signaling GCWARE (gcware) to terminate: [    OK    ]
Waiting for gcware services to unload:. [    OK    ]
Starting GCWARE (gcwexec): [    OK    ]
```

```
Starting GCMonit success!
Starting GBASED       : [    OK    ]
Starting GCLUSTERD :      [    OK    ]
Starting GCSYNC : [    OK    ]
Starting GCRECOVER : [    OK    ]
$ gccli -uroot
GBase client 9.5.3.17.117651. Copyright (c) 2004-2019, GBase.    All Rights Reserved.
gbase> show    priorities    where node_ name = 'node1';
+-----------+-------+----------+-----------------+--------+-------------+
| node_ name | group | priority | priority-weight | status | description|
+-----------+-------+----------+-----------------+-------+-------------+
| node1        |         0|            0 |                    20 | OFF       |.....           |
| node1        |         0|            1 |                    40 | OFF       |.....           |
| node1        |         0|            2 |                    60 | OFF       |.....           |
| node1        |         0|            3 |                    80 | OFF       |.....           |
| node1        |         1|            0 |                    20 | OFF       |.....           |
| node1        |         1|            1 |                    40 | OFF       |.....           |
| node1        |         1|            2 |                    60 | OFF       |.....           |
| node1        |         1|            3 |                    80 | OFF       |.....           |
| node1        |         2|            0 |                    20 | OFF       |.....           |
| node1        |         2|            1 |                    40 | OFF       |.....           |
| node1        |         2|            2 |                    60 | OFF       |.....           |
| node1        |         2|            3 |                    80 | OFF       |.....           |
.....
| node1        |        15|            0 |                    20 | OFF       |.....           |
| node1        |        15|            1 |                    40 | OFF       |.....           |
| node1        |        15|            2 |                    60 | OFF       |.....           |
| node1        |        15|            3 |                    80 | OFF       |.....          |
+-----------+-------+----------+-----------------+-------+-------------+
64 rows in set
```

- Example 5: Restart the cgroup configuration service (service cgconfig start) of node1.

```
# service cgconfig start
```

```
Starting cgconfig service: [    OK    ]
# su - gbase
# gcluster_ services all restart
Stopping GCMonit success!
Signaling GCRECOVER (gcrecover) to terminate: [    OK    ]
Waiting for gcrecover services to unload:... [    OK    ]
Signaling GCSYNC (gc_sync_server) to terminate: [    OK    ]
[    OK    ]for gc_ sync_ server services to unload:[    OK    ]
Signaling GCLUSTERD    to terminate: [    OK    ]
.[    OK    ]or gclusterd services to unload:... [    OK    ]
Signaling GBASED    to terminate: [    OK    ]
.[    OK    ]or gbased services to unload:[    OK    ]
Signaling GCWARE (gcware) to terminate: [    OK    ]
Waiting for gcware services to unload:. [    OK    ]
Starting GCWARE (gcwexec): [    OK    ]
Starting GCMonit success!
Starting GBASED       : [    OK    ]
Starting GCLUSTERD :     [    OK    ]
Starting GCSYNC : [    OK    ]
Starting GCRECOVER : [    OK    ]

$ gccli -uroot

GBase client 9.5.3.17.117651. Copyright (c) 2004-2019, GBase.    All Rights Reserved.

gbase> show    priorities    where node_ name = 'node1';
+-----------+-------+----------+----------------+--------+-------------+
| node_ name | group | priority | priority-weight | status | description|
+-----------+-------+----------+----------------+-------+-------------+
| node1        |        0|            0 |                  20 | ON        |.....          |
| node1        |        0|            1 |                  40 | ON        |.....          |
| node1        |        0|            2 |                  60 | ON        |.....          |
| node1        |        0|            3 |                  80 | ON        |.....          |
| node1        |        1|            0 |                  20 | ON        |.....          |
| node1        |        1|            1 |                  40 | ON        |.....          |
```

```
| node1     |      1|       2 |                60 | ON    |......     |
| node1     |      1|       3 |                80 | ON    |......     |
| node1     |      2|       0 |                20 | OFF   |......     |
| node1     |      2|       1 |                40 | OFF   |......     |
| node1     |      2|       2 |                60 | OFF   |......     |
| node1     |      2|       3 |                80 | OFF   |......     |
......
| node1     |     15|       0 |                20 | OFF   |......     |
| node1     |     15|       1 |                40 | OFF   |......     |
| node1     |     15|       2 |                60 | OFF   |......     |
| node1     |     15|       3 |                80 | OFF   |......     |
+-----------+-------+----------+-----------------+-------+-------------+
64 rows in set
```

6. **Configure priority queue related parameters**

   Complete the configuration of the priority queue by modifying the following parameter values in the configuration files of gcluster ($GCLUSTER_BASE/configure/gbase_8a_gcluster. cnf) and gnode ($GBASE_BASE/configure/gbase_8a_gbase. cnf):

   - gbase_ use_ priority_ queue:

     Set the parameter to 0 to close the priority queue;

     Set to 1 to enable priority queues.

   - _ gbase_ priority_ total_ Tasks: The parameter represents the maximum number of parallel running query tasks, including the query part of DML. The maximum value of this parameter cannot exceed 128, and the default is twice the number of local CPU cores;
   - _ gbase_ priority_ Tasks: The parameter represents the maximum number of tasks that can be accommodated in each priority queue (i.e. the number of tasks that can participate in scheduling). Tasks that fail to enter the queue will block waiting. The maximum value of this parameter cannot exceed 64, and the default is the number of local CPU cores;
   - gbase_ use_ res_ ctrl_ Group: This parameter determines whether to enable resource control group hooking. The parameter is set to 0, indicating that the resource control group connection is not enabled, and the default setting is not enabled; Set other values to enable resource control group hooking.

7. **Specify query SQL priority**

The resource group user Session can determine the SQL running level (i.e. the corresponding priority) based on the specific situation by using hint (format:/*+PRIORITY ('priority_value ') */). This command is only limited to querying SQL.

- Grammar:

```
Select /*+PRIORITY(‘priority_value’)*/ …
```

Permission requirements

Users with create, insert, drop, and select permissions.

- remarks:

The priority setting can only be less than or equal to the priority level of the user. If the setting is incorrect, it will revert to the user priority and issue a warning 'can not upgrade to priority X' (X is the priority level of the executing user).

- Example:

```
gbase> create table t1(a int);
Query OK, 0 rows affected

gbase> insert into t1 values (1),(1),(2),(3),(5);
Query OK, 5 rows affected
Records: 5  Duplicates: 0  Warnings: 0

gbase> select  /*+PRIORITY( ‘0’ )*/ * from t1 group by a;
+------+
| a    |
+------+
|    1 |
|    2 |
|    3 |
|    5 |
+------+
4 rows in set
```

5.3.2.2 **Load balancing strategy**

The GBase 8a MPP Cluster product supports load balancing strategies. There are three levels of support capabilities:

1. During the connection phase of the client application to the cluster, the node with the lowest current load is automatically selected for connection.

- ADO.NET:

```
String _ ConnString =
“server=192.168.0.2;failover=true;iplist=192.168.0.3;192.168.0.4;gclusterid=g1”
```

- C API:

```
Host=” 192.168.1.1; 192.168.1.2”
```

- JDBC:

```
String URL=”jdbc: gbase://192.168.1.56/test?user=gbase&password= ******&failoverEnable
=true&hostList=192.168.1.57, 192.168.1.58&gcluster=gcl1”
```

- ODBC：

```
“DRIVER=GBase 8a MPP Cluster ODBC 8.3 Driver;UID=gbase;PWD=******;”
“SERVER={192.168.111.96; 192.168.5.212; 192.168.7.174; 192.168.7.173};”
“CONNECTION_BALANCE=1;GCLUSTER_ID=gcluster;”
“CHECK_INTERVAL=90;”
```

2. In terms of data distribution strategy, it supports a uniform distribution strategy to ensure that the data volume of each node is uniform.
3. In the SQL execution distribution strategy, requests are decomposed into parallel execution on each host, ensuring that the load on each host is close to consistency.

5.3.2.3 **Compression strategy**

The performance bottleneck in most applications is disk I/O, so the main design goal of new databases is to reduce disk I/O. Data compression can reduce I/O time and improve performance, and GBase 8a MPP Cluster is no exception. Compression is also one of the main technologies to improve performance. The GBase 8a MPP Cluster parallel executor has been able to schedule decompression from the upper layer in parallel, greatly improving the applicability of decompression, In many scenarios, especially for scenarios with large amounts of data, using compressed data can achieve better performance than not compressing it.

5.3.2.4 **Expansion and contraction optimization**

- The highest value of the Gnode configuration parameter in the case of scaling:

```
MAX_ PARALLEL_ DEGREE = ( PROCESS_COUNT > ((TOTAL_NODES_COUNT-1) // (NEW_NODE_COUNT)) ? PROCESS_ COUNT / ((TOTAL_NODES_COUNT-1) // (NEW_NODE_COUNT)) : 1）;
```

It can avoid memory shortage and error reporting caused by configuration issues during scaling.

- RESULT_ BUFF_ COUNT=(Number of reserved nodes/Number of nodes removed from the group) * MAX_ PARALLEL_ DEGREE;

Among them:

- PROCESS_ COUNT: Number of CPUs;
- TOTAL_ NODES_ COUNT: The total number of nodes in the cluster;
- NEW_ NODE_ COUNT: The number of nodes reduced or increased in the cluster;

  Formula for maximum configured memory:
- RESULT_ BUFF_ COUNT * gbase_ buffer_ Result+other heap memory configuration parameters (data heap, temp heap)<80% of physical memory

If parallelism is enabled, then:

- TableParallel=The default number of running node CPUs, which is the set value after setting.

  Formula for maximum configured memory:
- TableParallel * gbase_ buffer_ Result+other heap memory configuration parameters (data heap, temp heap)<80% of physical memory.

- Stable query performance during node replacement in scaling+expansion mode

  Scenario of node replacement using scaling and expansion methods, enabling parameter gcluster_ different_ distribution_ After optimization, the performance of cross distribution queries during node replacement does not decrease.

  A value of 0 represents off, a value of 1 represents on, and the default value is 0.

  explain:

  During the node replacement process using the scaling and expansion method, the cluster will have three distribution IDs: the original distribution ID, the scaled ID, and the scaled ID. The data distribution of the three distribution IDs remains unchanged. If there are associated queries during this process that involve tables that have been resized and redistributed, or have not been completed, causing these tables to be on different distribution IDs, then:

  When this parameter is turned off, the execution plan will perform a pull table

operation on queries across distribution IDs, pulling the table data to the same distribution ID before executing the operation;

After the parameter is turned on, the execution plan will check the data distribution on different distribution IDs. If the data distribution of the table is the same, it will be executed according to the same distribution ID. This will reduce the pull table operation and ensure that the query performance during node replacement reaches the performance before replacement.

5.3.2.5 Optimization of Asynchronous Dual Active Cluster Data Synchronous Transmission

The following optimizations are limited to data synchronization between two clusters, and the synchronization of active and standby data within the cluster is not affected

1) Data synchronization speed limit:

The total network bandwidth limit that can be used for data synchronization between two clusters.

Setting the bandwidth limit too small can cause an error prompt. The total bandwidth setting must be greater than 1MB/s, and after calculation, the synchronization bandwidth of each partition must not be less than 10KB/s.

Each shard bandwidth limit=Total bandwidth limit/Table parallelism/Number of synchronized shards

The number of synchronized shards is affected by rsync_ Mode Impact

2) Compressed transmission:

The metadata and DC data files that need to be synchronized between two clusters are compressed twice using the zlib algorithm.

If the data is less than 50Byte, no secondary compression will be performed; If the data is greater than 50Byte but the compressed data length is greater than the pre compressed data length, no secondary compression will be performed.

- Optimize opening operation:

  1) Add the following parameters to the configuration file synctool. conf:

  BANDWIDTH_ QOS_ SWITCH=1

  COMPRESSED_ SWITCH=1

  Note: All parameters in the configuration file need to be capitalized. The default

values for the above two parameters are 0, indicating that the function is not enabled, and a value of 1 indicates that the function is enabled.

2) Add parameters when starting the synchronization tool:

./gcluster_ rsynctool.py --sync_ network_ Bandwidth=bandwidth limit

Explanation: The default unit for the bandwidth limit is MB/s, with a default value of 0, indicating that the bandwidth is not limited. The value range is 1~99999.

- Performance reference:

  The compression ratio of secondary compression shall not be less than 70%, and the data transmission time after secondary compression shall not increase by more than 50% compared to non compression

- Compatibility note:

  The optimization function has made changes to the synchronous transmission protocol, so the version of the dual active cluster that enables the optimization function is not compatible with the old version.

  By using the optimization function, both active and inactive clusters must use sync that includes this feature_ Client and sync_ The corresponding version of the server.

  When the optimization function (compression function and current limiting function) is turned off, there is no requirement or impact on the dual active cluster version.

5.3.3 Optimization instance

5.3.3.1 DML optimization

5.3.3.1.1 Association optimization

5.3.3.1.1.1JOIN Association Optimization Strategy

- Check the right table one by one. If the right table is a distributed table and breaks the hash distribution, if the data volume is small, directly modify it to a replicated table to avoid pulling large tables; If the data volume is large (with over 100 million records), adjust the gcluster_ hash_ redistribute_ join_ Optimize parameter validation, parameter settings
- The explanation is as follows:
- 0-- Pull replication table

- 1- Redistribution
- 2- Automatic evaluation, based on the amount of data, the number of rows in the left and right tables is close to using redistribution. If the difference is large, the small table will be pulled and copied from the table.

be careful

When encountering a left join statement, the principle for evaluating whether the right table is set to replication is:

- If the number of fields in the right table is no more than 10 and the number of records is no more than 50 million rows, then the right table is created as a copied table;
- If the number of fields in the right table is greater than 10 and the number of records is not greater than 10 million, the right table is created as a replicated table.

5.3.3.1.1.2Optimization of association order

1. **Optimization reasons**

   The optimizer of GCluster does not adjust the order of the Left JOIN statements, and the JOIN order of user statements may not be optimal, resulting in lower query performance.

2. **SQL Features**

   The statement contains multiple Left JOINs, and the ON condition for multiple Left JOINs is t1. colX=tn. colX

   For example:

```
SELECT x1.* FROM x1
LEFT JOIN x2 ON x1.many_ duplicate_ value = x2.many_ duplicate_ value
LEFT JOIN x3 ON x1.no_ duplicate_ value = x3.no_ duplicate_ value
LEFT JOIN x4 ON x1.hash_ col = x4.hash_ col;
```

3. **Optimize Scenarios**

   The statement features meet the above feature description.

   The right table of the Left JOIN, some tables can directly form a Hash JOIN relationship with the left table, and some tables may cause the left table to expand.

4. **Optimization effect**

   Let the Left JOIN that forms a Hash JOIN relationship execute first to avoid pulling tables.

   For example, the statement described in SQL features, because left join x4 on

x1.hash_ col = x4.hash_ Col is a Hash distributed JOIN, so it can be mentioned at the beginning and executed directly in a distributed manner.

Let the left JOIN with low expansion rate execute first to reduce the amount of pull table data.

If the repeatability of the values of the columns participating in the JOIN condition is high, it is likely to cause the left JOIN result to expand. Generally speaking, when using JOIN conditions involving primary key columns, the inflation rate is the smallest; The columns with more duplicate values are more likely to have higher expansion rates.

For example, the statement described in SQL features, due to left join x3 on x1.no_ duplicate_ value = x3.no_ duplicate_ Value to x1 expansion ratio left join x2 on x1.many_ duplicate_ value = x2.many_ duplicate_ The value is small, so left join x3 can be mentioned before left join x2.

Through this adjustment, it avoids pulling tables on the inflated data and reduces the amount of data pulled.

Example statement:

```
SELECT x1.* FROM x1
LEFT JOIN x2 ON x1.many_ duplicate_ value = x2.many_ duplicate_ value
LEFT JOIN x3 ON x1.no_ duplicate_ value = x3.no_ duplicate_ value
LEFT JOIN x4 ON x1.hash_ col = x4.hash_ col;
```

Rewritten statement

```
SELECT x1.* FROM x1
LEFT JOIN x4 ON x1.hash_ col = x4.hash_ col
LEFT JOIN x3 ON x1.no_ duplicate_ value = x3.no_ duplicate_ value
LEFT JOIN x2 ON x1.many_ duplicate_ value = x2.many_ duplicate_ value;
```

explain

- Due to x1.ash_ col = x4.hash_ Col uses Hash distribution columns, so the left join x4 is adjusted to the first position;
- Due to x1.no_ duplicate_ value = x3.no_ duplicate_ The expansion ratio of value is x1.many_ duplicate_ value = x2.many_ duplicate_ The expansion rate of value is low, so adjust left join x3 to before left join x2.

5.3.3.1.1.3Optimization of association conditions

1. **Case 1- Optimization of the order of association conditions**

- **Optimization reasons**

  In the current version of GNode, the right form table condition in the ON condition of the Left JOIN will be executed after the JOIN, especially when the right table of the Left JOIN is a large table, which will result in an excessive amount of data participating in the JOIN and increase JOIN time.

- **SQL Features**

  Left JOIN statement

  Single table condition with right table in ON condition

- **Optimize Scenarios**

  The right table of the Left JOIN is a large table

  In the ON condition, the filtered data volume of the single table condition in the right table accounts for relatively little (about 10%) of the total data volume in the right table.

  Note: Because this optimization rewrite involves rewriting the right table as a subquery, it is necessary to consider the additional materialized consumption of the subquery. Therefore, not all such SQL rewrites can improve performance, especially when there are a large number of right table columns in the projected columns of the query.

- **Optimization effect**

  By rewriting, the filtering of the right form table is placed in an independent sub query to ensure that the filtering of the right table is executed before JOIN, achieving the goal of optimizing query performance.

  Example statement:

```
SELECT x1.id2, x2.id2, x2.id3 FROM x1 LEFT JOIN x2 on x1.id2 = x2.id2 AND x2.id3 = 301;
```

  Rewritten statement

```
SELECT x1.id2, x2.id2, x2.id3 FROM x1 LEFT JOIN (SELECT x2.id2, x2.id3 FROM x2 WHERE x2.id3 = 301) x;
```

2. **Case 2- Optimization of Associated Conditions with Subqueries**

- **Optimization reasons**

  In GNode, the related sub queries in the ON condition of the Left JOIN cannot be

executed in an optimized manner and need to be executed in a row by row substitution method, which leads to extremely low execution performance of the related sub queries.

- **SQL Features**

  Left JOIN statement

  The ON condition contains a single table condition for the right table, and this single table condition is a related subquery

- **Trace Information**

  When related sub queries cannot be executed in an optimized manner, the GNode's trace will contain the following information:

```
can't optimize this subselect because OUTER JOIN or "outer select's table is used in having" or ... !
```

- **Sample statement**

```
SELECT x1.id2, x2.id2, x2.id3 FROM x1 LEFT JOIN x2 on x1.id2 = x2.id2 AND EXISTS (SELECT 1 FROM x3 WHERE x3.id3 = x2.id3);
```

Rewritten statement

```
SELECT x1.id2, x.id2, x.id3 FROM x1 LEFT JOIN (SELECT id2, id3 FROM x2 WHERE EXISTS (SELECT 1 FROM x3 WHERE x3.id3 = x2.id3)) x ON x1.id2 = x.id2;
```

3. Case 3- Improving Performance by Adding Associated Condition Fields

When calculating intra table associations in SQL statements, especially when the data volume of the table exceeds tens of millions, you can try to create the association calculation as a field and make direct judgments based on the results of the new field.

Example:

```
select
        …
        …
from rep.statcmain a ,rep.statcitemkind b;
```

Where table association conditions

and a.statdate <= a.endstatdate

and ..

The amount of data in the rep.statcmain table is 81864314. When comparing the sizes of a.statdate and a.endstatdate for intra table correlation calculations, it takes a

long time and is modified to the following two steps for optimization,

First, create a numerical field named statdate on rep.statcmain_ endstatdate。

Enable statdate_ endstatdate= a.statdate- a.endstatdate。

update rep.statcmain set statdate_ endstatdate = cast(statdate as date) - cast(endstatdate as date);

When associating tables, use statdate_ The endstatdate field filters data.

```
 select
      …
      …
from rep.statcmain a ,rep.statcitemkind b,
```

Where table association conditions

and statdate_ endstatdate <= 0

and ..

Performance comparison: 1 hour and 50 minutes before optimization, and 45 minutes after optimization.

5.3.3.1.1.4Specify the join order through hint

Grammar:

/*+ join_ path('tablename,tablename[,...]')*/

/*+ join_ path('(tablename,tablename),[tablename][,(tablename,tablename[,...])]*/

Add the join order specified by hint to provide a basis for generating query plans.

Example:

join_ path('(a,b),(c,d)')

Join Table a and Table b first, join Table c and Table d, and then join the two results

join_ path('a,(b,c)')

Join Table b and Table c, then join Table a

join_ path('a,((b,c),d))

Join Table b and Table c first, then join with Table d, and finally join Table a with the previous results

join_ path('(a,b,c),(d,e)')

Join Table a, Table b, and Table c in writing order, join Table d and Table e, and join the result set of the last two

explain:

1. This function is subject to parameters_ t_ gcluster_ user_ defined_ join_ Hint control, with a default value of 0 indicating that the function is turned off and 1 indicating that it is turned on.

2. To use this hint function, all table or subquery aliases involved in the from clause used in the current select clause must be specified. Table names or subquery aliases that do not exist in the from clause cannot be specified in this hint.

3. Cannot change SQL semantics. For example, it is

not allowed to change the connection order in Hint for external and internal connections or external and external connections; It is not allowed to change the connection order in Hint for two inner connections with an outer connection in between.

4. If there is an abnormality in the join order specified by hint, simply ignore hint and record the warning information in the log file. The recorded information is as follows:

Error number	meaning
JOINPATHERR001	Syntax error
JOINPATHERR002	The hint specifies that there is no table name or subquery alias present
JOINPATHERR003	Incomplete table name or subquery alias specified in hint
JOINPATHERR004	Involving the exchange sequence of external connections or cross external connections

5. This function is only valid for the express engine table and is only optimized for clusters. It does not involve single machines and does not affect single machine functionality, meaning that the specified join order is reflected in the plan generated at the cluster level. You can view the hint through explain:

explain extended select /*+ join_ path('t3,t2,t1') */ * from t1,t2,t3 where t1.a=t2.a and t3.a=t1.a;

5.3.3.1.1.5Union Optimization of Cluster Execution Plan

Union Execution Method

There are three inherent ways for cluster planning to execute a union:

1. Direct distribution: can perform union operations on various nodes, and the results do not need to be summarized and directly returned to the client

For example:

select id4 from x1 union select id4 from x2;

X1 and x2 are hash distribution tables, and id4 of both tables is the hash distribution column

select entry_ id,id4 from x1 minus select entry_ id,id2 from r1;

X1 is the hash distribution table, id4 is the hash distribution column of x1, and r1 is the replication table.

2. Single node execution: Only one node needs to execute and return results

For example:

select * from r1 union select * from r2;

R1 and r2 are replicated tables

select entry_ id from x1 where x1.id4=10 union select id2 from x2 where x2.id4=10;

X1 and x2 are hash distribution tables, and the distribution columns of both tables are id4

3. Execute first and then summarize: Each node executes a union first, and the results are summarized on a single node before executing distinct on the summary node.

Union optimization principle:

This optimization is controlled by parameters, mainly by increasing the number of computing nodes. When there are more than 10 nodes, the performance improvement is significant. The specific optimization method is as follows:

If it can meet the requirements of "direct distribution" and "single node execution", execute according to "direct distribution" and "single node execution"; For those that cannot be met, they can be converted to "direct distribution" or "single node execution" by pulling a table for execution, with priority given to conversion to direct distribution for execution. Pull table includes pull replication table and hash redistribution pull table.

Union optimization control parameters

- _ t_ gcluster_ union_ redist_ optimize

Control whether to enable union redistribution optimization, with a value range of [0,1,2]

The default value is 0, which means it is off

1 indicates that it is enabled and does not support redistribution optimization with

int and decimal columns corresponding to both sides of the union

2 indicates that it is enabled and supports the redistribution optimization of int and decimal columns corresponding to both sides of the union. Before redistribution, the int column will be converted to decimal type for redistribution

Note:

Hash redistribution refers to the redistribution of data in a table using temporarily specified columns during union operations as hash distribution columns. Different data types have different hash algorithms, and union optimization requires that the hash column data types at the corresponding positions of each clause be the same. If they are not the same, they need to be converted to consistent data types. Currently, only data type conversions with int ->decimal are supported, and conversions between other types are not supported.

- _ t_ gcluster_ union_ redist_ distinct

Control whether to include distinct when pulling tables in sub queries.

Value range [0,1]

The default value is 1, indicating a distinct

Note:

For cases where deduplication is required, when a union subquery pulls a table, the distinct keyword is usually added to the projection column to perform deduplication first to reduce the amount of data to be pulled and calculated in the next step. When the repetition rate of sub query data is low and the amount of data required for a single distinct reduction is not significant, this parameter can be set to 0 to reduce the distinct once.

5.3.3.1.1.6Left Join optimized configuration

Combinatorial optimization of left join right form table condition push down and temporary table reuse

- _ t_ gcluster_ push_ down_ cond_ of_ right_ table_ for_ left_ join

Function: Control whether the left join right form table condition push down optimization function is enabled

1: Open

0: Close

Default value is 1, global/session parameter

- _ t_ gcluster_ reuse_ tmp_ table_ optimize

Function: Control whether the temporary table reuse function is enabled

0: Close

1: Enable temporary table reuse

2: Enable reuse of from subqueries

Default value is 0, global/session parameter

- Example of combinatorial optimization:

_ t_ gcluster_ push_ down_ cond_ of_ right_ table_ for_ left_ Join=1 (default value is 1)

_ t_ gcluster_ reuse_ tmp_ table_ Optimize=2 (default value is 0)

Left Join Right Form Table Condition Pushdown Optimization Does Not Affect Temporary Table Reuse Function

Example:

Execution plan in default configuration:

```
gbase> create table t1(a int,b int, c int) distributed by('a');
Query OK, 0 rows affected (Elapsed: 00:00:00.18)

gbase> create table t2(a int,b int,c int) distributed by('a');
Query OK, 0 rows affected (Elapsed: 00:00:00.07)

gbase> create table t3(a int,b int,c int) distributed by('a');
Query OK, 0 rows affected (Elapsed: 00:00:00.04)

gbase> desc select 1 from t1 left join t2 on t1.a=t2.b and t2.c>1 and t2.a in(select a from t3) union select 1 from t1 left join t2 on t1.a=t2.b and t2.c>1 and t2.a in(select a from t3);
+----+-------------+-----------+-------+--------------------
+
| ID | MOTION          | OPERATION    | TABLE | CONDITION                |
+----+-------------+-----------+-------+--------------------
+
| 03 | [RESULT]       |   Step          | <02>   |                          |
|      |                  |   AGG           |          |                          |
| 02 | [GATHER]       |   LEFT JOIN |          | (a = b)                |
|      |                  |    Table       | t1[a] |                          |
```

```
|    |                  |    Step        | <00>   |                          |
|    |                  |  UNION         |        |                          |
|    |                  |  LEFT JOIN |           | (a = b)                  |
|    |                  |    Table       | t1[a] |                           |
|    |                  |    Step        | <01>   |                          |
| 01 | [REDIST(b)] |   SCAN          | t2[a] | (c{S} > 1)               |
|    |                  |                |           | a IN ([SubQuery3]) |
| 00 | [REDIST(b)] |   SCAN          | t2[a] | (c{S} > 1)               |
|    |                  |                |           | a IN ([SubQuery1]) |
+----+-------------+-----------+-------+--------------------
+
```

Execute plan after adjusting parameters:

```
gbase> set _ t_ gcluster_ reuse_ tmp_ table_ optimize=2;
gbase> show variables like '%_ t_ gcluster_ push_ down_ cond_ of_ right_ table_
for_ left_ join%';
+-----------------------------------------------------------+--
-----+
| Variable_ name                                                | Value |
+-----------------------------------------------------------+--
-----+
| _ t_ gcluster_ push_ down_ cond_ of_ right_ table_ for_ left_ join | 1         |
+-----------------------------------------------------------+--
-----+
gbase> show variables like '%_ t_ gcluster_ reuse_ tmp_ table_ optimize%';
+--------------------------------------+-------+
| Variable_ name                           | Value |
+--------------------------------------+-------+
| _ t_ gcluster_ reuse_ tmp_ table_ optimize | 2         |
+--------------------------------------+-------+
```

```
gbase> desc select 1 from t1 left join t2 on t1.a=t2.b and t2.c>1 and t2.a in(select a
from t3) union select 1 from t1 left join t2 on t1.a=t2.b and t2.c>1 and t2.a in(select
a from t3);
+----+-------------+-----------+-------+--------------------
```

```
+
| ID | MOTION          | OPERATION    | TABLE | CONDITION                |
+----+-------------+-----------+-------+-------------------
+
| 02 | [RESULT]       |   Step          | <01>   |                          |
|      |                  |   AGG           |          |                          |
| 01 | [GATHER]       |   LEFT JOIN |          | (a = b)                  |
|      |                  |     Table       | t1[a] |                          |
|      |                  |     Step        | <00>   |                          |
|      |                  |     Step        | <00>   |                          |
|      |                  |   UNION         |          |                          |
|      |                  |   LEFT JOIN |          | (a = b)                  |
|      |                  |     Table       | t1[a] |                          |
|      |                  |     Step        | <00>   |                          |
| 00 | [REDIST(b)] |   SCAN          | t2[a] | (c{S} > 1)               |
|      |                  |                 |          | a IN ([SubQuery1]) |
+----+-------------+-----------+-------+-------------------
+
```

5.3.3.1.2 Query condition optimization

5.3.3.1.2.1Do not use functions for query criteria

It is recommended to avoid using function operations on columns in the where condition as much as possible, as adding function operations can cause smart index failure and reduce SQL performance.

- For example:

  The original SQL is:

```
where substr(product_no, 2, 1) in ('3', '4', '5', '8')
```

  Intelligent index failure, very low performance.

  Rewrite as:

```
where (product_no like '13%' or product_no like '14%' or product_no like '15%' or product_no like '18%')'
```

  The smart index will index the first 8 characters of string type data.

5.3.3.1.2.2Avoid comparing after manipulating field expressions

1. The most effective way to use intelligent indexing is through direct manipulation of fields and constant expressions:

```
(rownumtag>=100*10);
```

Change to

```
(rownumtag+1>=100*10+1);
```

You cannot use smart indexes.

2. Secondly, one side is a field, and the other side is a constant expression (which can also be a constant). No matter how complex a constant expression is, it is not a problem because it only needs to be evaluated once.

The condition for comparing an expression with a constant cannot be used with a smart index.

For example:

```
SELECT ... FROM ... WHERE ceil(rownumtag / ceil(TO_NUMBER('100')))
='10' ;
```

Change to:

```
SELECT ... FROM ... WHERE rownumtag>100*9 AND    rownumtag<=100*10;
```

5.3.3.1.3 Union all issues

For the scenario of union all, the overall idea is to establish a temporary table to put various SQL results into the temporary table, and finally query the temporary table for unified output.

5.3.3.1.4 Update operation

When modifying multiple updates to associate updates, be sure to place non updated tables after the update.

```
UPDATE t1, t2 SET t2.col = 1 WHERE t1.id = t2.id;
```

As stated above, t1 is the associated table and t2 is the update table. In the updated JOIN list, the associated table t1 should be written before the updated table t2, which helps the GCluster layer reduce pull table actions.

be careful

Do not write in the following form:

```
update t2, t1 set t2.col = 10 where t1.id = t2.id;
```

The update analyzer is not optimized and will default to pulling the left table. If the update table is written before it, it will default to pulling it out and updating it before putting it back, taking an extra step.

5.3.3.1.5 Change the loop insert values to an insert select

Change the scenario of executing insert values in a loop to an insert select statement, which avoids multiple submissions and improves performance significantly.

5.3.3.1.6 Using rows and columns to reduce node I/O

GBase 8a MPP Cluster provides row column mixed storage function, which improves I/O performance by storing redundant row information. When the number of statistical columns involved exceeds 15-20%, the performance of using row storage columns is better, including aggregation columns and grouping columns.

be careful

1. Establishing rows and columns will double the space redundancy.
2. Loading performance will also decrease.

5.3.3.1.7 Rewriting the MAX OVER function

Example 1:

```
SELECT
MIN(kpi_value) OVER (PARTITION BY a.kpi_id,a.brand_id,a.city_id ORDER BY FLOOR( DAYS(a.kpi_date) ) RANGE BETWEEN 1 preceding AND 1 preceding ) as last_ value
FROM a;
```

Rewrite as:

```
SELECT
(SELECT MIN(kpi_value) FROM kpi_ values aa WHERE aa.kpi_ id = a.kpi_ id
AND aa.brand_ id = a.brand_ id AND aa.city_ id = a.city_ id AND aa.kpi_ date =
DATE_ SUB(a.kpi_date, INTERVAL 1 DAY)) as last_ value FROM a ORDER
BY a.kpi_ id,a.brand_ id,a.city_ id,a.kpi_ date;
```

Example 2

```
SELECT MAX(rn) OVER (PARTITION BY product_no,imei ORDER BY
first_time) rn FROM a;
```

Rewrite as:

```
SELECT
(SELECT MAX(rn) FROM a aa WHERE aa.product_ no = a.product_ no AND
aa.imei=a.imei AND aa.first_ time <= a.first_ time)
FROM a ORDER BY a.product_ no,a.imei,a.first_ time;
```

5.3.3.2 **DDL optimization**

For the cluster layer, it mainly refers to setting the distribution attributes of the table reasonably and selecting the hash distribution columns reasonably. For GNode layers, it mainly refers to compression parameters.

5.3.3.2.1 Copy Table Optimization Scheme

When encountering multi table associations, especially when multiple right tables are connected outside the main table, and the association field is the hash key of the right table, in order to prevent breaking the hash distribution calculation, it can be created as a replicated table based on the size of the right table's data volume.

give an example:

```
SELECT
..
..
From rep.statcmain a --80989472 hash column policyno
INNER JOIN rep.statdcompanylevel d        --25887    replicate
ON    a.comcode = d.comcode
LEFT JOIN rep.statdagent    l            --86485    replicate
ON a.agentcode = l.agentcode
LEFT JOIN rep.temp_ Prpcengagenew pr --164205 hash column policyno
ON a.policyno = prpcengagenew.policyno
LEFT JOIN rep.statdcarmodel      b          --178758 replicate
ON a.modelcode = b.modelcode
LEFT JOIN rep_ dev.odsbi_ prpmotorcade i    --288949 replicate
ON a.contractno = i.contractno
LEFT JOIN ..
```

Among them, the main table rep.statcmain has a data volume of 80989472, hash column policyno, but the external connection tables rep.statdcompanylevel, rep.statdagent, rep.statdcarmodel, rep_ dev.odsbi_ The associated fields of prpmotorcade are non hash keys. Creating these tables as replicated tables can prevent pull tables from being distributed in one step.

5.3.3.3 DQL optimization

5.3.3.3.1 first_ Rows optimization

explain

Using first_ Rows optimization needs to be enabled through hint mode and first_ The rows keyword is issued to the Data node for execution. first_ Rows optimization can prompt the Data node to immediately send the result set with a specified number of rows to the client and output it on the client after completing the materialization.

be careful

- Use the - c and - q parameters when logging into the cluster
  - -The c parameter ensures that hint (i.e./*+... */) is not directly ignored by the client and is sent to the server.
  - -The q parameter immediately displays the query result set on the client.
- For single table queries, if the limit keyword is used, it is required that the limit cannot contain offset.
- The query statement cannot contain GROUP BY, ORDER BY, or OLAP functions. UNION is not supported, but UNION ALL is supported.

Grammar format

```
SEELCT /*+ first_ rows(n) */ columns FROM
[vc_name.][database_name.]table_ name LIMIT n;
```

Table -5166 Parameter Description

Parameter Name	explain
vc_ name	VC name, optional.
database_ name	Database name, optional.
table_ name	Table Name
n	Represents the request to return the smallest result set each time.

Example

Example:/*+first_ rows(5) */ t1.a

```
gbase> SELECT /*+ first_ rows(5) */ t1.a FROM t1 LIMIT 10;
+------+
| a    |
+------+
|    1 |
|    2 |
|    3 |
|    4 |
|    5 |
|    6 |
|    7 |
|    8 |
|    9 |
|   10 |
```

```
+------+
10 rows in set
```

5.3.3.3.2 Optimization of Equivalent HASH Related Subqueries

explain

When the table involved in the query has a Hash distributed column and the query condition is equivalent to that column, this function (supporting consistent Hash and static Hash of node count) can be used to send the query only to the specified Hash node to reduce the connection consumption of nodes participating in the operation, as well as coordinator nodes and data nodes.

Using optimization methods

1) Set cluster configuration parameters gcluster_ special_ correlated_ Optimize=0 | 1, which is used to set whether to attempt dynamic redistribution optimization on related subqueries. A parameter value of 0 indicates that this optimization is turned off, while a parameter value of 1 indicates that this optimization is turned on, which is enabled by default.

2) Set cluster configuration parameters gcluster_ crossjoin_ use_ hash_ Distribution=0 | 1, used to set whether to still force the use of related subqueries for dynamic redistribution optimization when neither side of the JOIN is a hash column. A parameter value of 0 indicates that optimization is turned off, a parameter value of 1 indicates that optimization is turned on, and the default value is on.

explain

The above two parameters belong to the parent-child parameter relationship.

- When gcluster_ special_ correlated_ When optimize=0, regardless of setting the parameter gcluster_ crossjoin_ use_ hash_ The value of distribution will not enable this optimization.
- When gcluster_ special_ correlated_ Optimize=1, while gcluster_ crossjoin_ use_ hash_ When distribution=0, this optimization will not be enabled.
- When gcluster_ special_ correlated_ Optimize=1, and gcluster_ crossjoin_ use_ hash_ This optimization will only be enabled when distribution=1.

Example

Firstly, use the SET statement to set the values of both parameters to 1, and the following statement can be used for equivalent hash query optimization.

```
SELECT COUNT(*) FROM x1 WHERE EXISTS(SELECT 1 FROM x2 WHERE x1.id2 = x2.id2);
```

Use constraints

- In the query statement, cross layer relationships are used and optimization is prohibited.

  The following SQL statement will not be optimized because x3.id3=x1.id3 is a condition for cross layer related sub queries, so it cannot be optimized.

```
SELECT COUNT(*) FROM    x1 WHERE    EXISTS
(
    SELECT 1 FROM x2 WHERE    x2.id2 = x1.id2
    AND EXISTS
    (
        SELECT 1 FROM x3 WHERE    x3.id3 = x1.id3
    )
);
```

- When cross layer subqueries are used in the query statement and cross layer columns appear in the PROJECT section, optimization is prohibited.

  The following SQL statement will not be optimized because SELECT x1. entry_ In ID from x3, x1 and x3 belong to a cross layer relationship (not a parent-child relationship), and x1. entry_ The ID appears in the PROJECT section of the innermost subquery, so it cannot be optimized.

```
SELECT Count(*) FROM    x1
WHERE    EXISTS
(
    SELECT 1 FROM     x2
    WHERE    x2.id2 = x1.id2
    AND x2.entry_ id IN
    (
        SELECT x1.entry_ id FROM x3 WHERE    x2.id3 = x3.id3
    )
);
```

- In the query statement, related sub queries are in HAVING and optimization is prohibited.

  The following SQL statement will not be optimized because HAVING contains sub query statements.

```
SELECT    COUNT(*) FROM    x1 GROUP BY id2
```

```
HAVING EXISTS
(
    SELECT 1 FROM x2 WHERE    x2.id2 = x1.id2
);
```

- In the query statement, the relevant sub queries belong to the PROJECT section and optimization is prohibited.

  The following SQL statement will not be optimized because the sub query statement SELECT 1 from x2 WHERE x1.id2=x2.id2 LIMITED 1 belongs to the PROJECT section of the external query statement, so it cannot be optimized.

```
SELECT
(
    SELECT 1 FROM     x2 WHERE    x1.id2 = x2.id2 LIMIT    1
) FROM x1;
```

- In the query statement, there are two layers of related sub queries, and the outer layer cannot be optimized, resulting in both inner layers not being optimized.

  The following SQL statement will not be optimized because the outermost subquery statement SELECT 1 From x2 WHERE x1.id2>x2.id2 does not have an equivalent query condition, so it cannot be optimized.

```
SELECT COUNT(*) FROM x1
WHERE EXISTS
(
    SELECT 1 FROM x2
    WHERE x1.id2 > x2.id2
    AND EXISTS
    (
        SELECT 1 FROM x3 WHERE x3.id3 = x2.id3
    )
);
```

- In the query statement, the relevant condition is not of type out. col=inner. col (such as out. col>inner. col, or is not a simple field, but an expression, such as ABS (out. col)=inner. col, which does not support optimization).

  The following SQL statement will not be optimized because the function ABS (x1. id2)=x2. id2 is used in the sub query statement, so it cannot be optimized.

```
SELECT COUNT(*) FROM x1 WHERE EXISTS(SELECT 1 FROM x2
WHERE ABS(x1.id2) = x2.id2);
```

5.3.3.3.3 From sub query reuse optimization - using grouped hint to support multi column redistribution

When the following hint function is specified, the results of the from subquery are first saved to a temporary table and then used by subsequent queries.

Hint function name: grouped

Function: Specify the redistribution method of the result set

Value: [-10 | -2 | -1 | Projection column index list]

Default value: -10

explain:

-10 represents keeping the distribution attributes of the result set unchanged;

-2 represents the result set pull replication table;

-1 represents the RoundRobin distribution of the result set;

The projection column index starts from 0, so the projection column index list is represented as one or more values greater than or equal to 0, separated by commas, and the result set is hash redistributed according to the column specified by the column index. If any subscripts exceed the projection column range, hint is invalid.

select /*+grouped('-10')*/ ..... Sub query results are not redistributed and stored as is

select /*+grouped('-2')*/ ..... Pull and copy sub query results to a table

select /*+grouped('2')*/ ..... Redistribution of sub query results by hash in the third column of the projection column

select /*+grouped('2,3')*/ ..... Redistribution of sub query results by hash in columns 3 and 4 of the projection column

be careful:

When the from subquery is expanded and no longer exists, if hint is specified, it will not be expanded again

2. When the from subquery is pruned by the projection column, if hint specifies hash redistribution, pruning optimization will not be performed anymore

For example:

select * from (select /*+grouped('-2')*/ * from x1) xx, x2 where xx.id2=x2.id4;

If there is no hint, xx will first perform a hash redistribution and then join with x2. After hint specifies xx to pull the replication table, the plan will become xx to pull the replication table and then join with x2.

5.3.3.3.4 Limit... OFFSET performance optimization

explain

When the cluster encounters a simple query with LIMITED or "LIMITED... OFFSET", it will follow the optimization steps and not generate a summary table.

explain

- The definition of a simple query includes the following SQL statement scenarios:
  - The query is a single table query without sub queries;
  - The query has no DISTINCT, aggregate function or OLAP function;
  - There are no GROUP BY or ORDER BY clauses in the query;
  - Non SELECT INTO OUTFILE query.

Using optimization methods

The optimization strategy is to locate the position after LIMITED in the query results. Firstly, a COUNT (*) evaluation of the number of records that meet the conditions is performed on each data node to obtain the number of records that meet the conditions for each node. Then, based on the number of records that meet the conditions for each data node, the SQL is further organized and the query statement is accurately sent to the specified node for execution.

Example

```
SELECT * FROM t WHERE a > 0 LIMIT 1 OFFSET 2;
```

5.3.3.3.5 Partition filtering

explain

When querying the partition table, the missed partitions are automatically filtered out according to the query criteria, and only the hit partitions are queried

Using optimization methods

Partition filtering can be applied when the WHERE condition of the query SQL in the partition table or the WHERE condition collated by the GBase 8a MPP Cluster meets the following two conditions:

partition_ name = constant

partition_ name IN (constant1, constant2,……)

The WHERE condition can cover:<,<=,>,>=,=,<>, IN, BETWEEN AND... Etc.

Example

```
gbase> CREATE TABLE t1 (n INT) PARTITION BY RANGE(n)
```

```
        (
        PARTITION p0 VALUES LESS THAN (100),
        PARTITION p1 VALUES LESS THAN (200),
        PARTITION p2 VALUES LESS THAN (300),
        PARTITION p3 VALUES LESS THAN (400)
        );
Query OK, 0 rows affected (Elapsed: 00:00:00.13)

gbase> INSERT INTO t1 VALUES(1),(2),(3),(4),(5);
Query OK, 5 rows affected (Elapsed: 00:00:00.10)
Records: 5    Duplicates: 0    Warnings: 0

gbase> INSERT INTO t1 VALUES(101),(102),(103),(104),(105);
Query OK, 5 rows affected (Elapsed: 00:00:00.08)
Records: 5    Duplicates: 0    Warnings: 0

gbase> INSERT INTO t1 VALUES(201),(202),(203),(204),(205);
Query OK, 5 rows affected (Elapsed: 00:00:00.10)
Records: 5    Duplicates: 0    Warnings: 0

gbase> INSERT INTO t1 VALUES(301),(302),(303),(304),(305);
Query OK, 5 rows affected (Elapsed: 00:00:00.09)
Records: 5    Duplicates: 0    Warnings: 0

gbase> SELECT * from t1 WHERE n <4;
+------+
| n    |
+------+
|    1 |
|    2 |
|    3 |
+------+
3 rows in set (Elapsed: 00:00:00.05)

gbase> SELECT * from t1 WHERE n >300;
```

```
+------+
| n    |
+------+
|  301 |
|  302 |
|  303 |
|  304 |
|  305 |
+------+
5 rows in set (Elapsed: 00:00:00.03)

gbase> SELECT * FROM t1 WHERE n IN (101,309);
+------+
| n    |
+------+
|  101 |
+------+
1 row in set (Elapsed: 00:00:00.04)

gbase> SELECT * FROM t1 WHERE n BETWEEN 1 AND 199;
+------+
| n    |
+------+
|    1 |
|    2 |
|    3 |
|    4 |
|    5 |
|  101 |
|  102 |
|  103 |
|  104 |
|  105 |
+------+
10 rows in set (Elapsed: 00:00:00.05)
```

5.3.3.4 **Business tuning**

Business tuning is the most complex, and only by becoming familiar with and mastering the application business processes can we find the correct optimization points and methods. Business optimization must be fully coordinated with on-site personnel to ensure that the optimized business logic is equivalent to the pre optimized business logic.

5.3.3.4.1 Performance optimization of zipper watch

Optimize Scenarios

Long term data accumulation leads to a discrete distribution of data in a table with enddate='29991231', which is stored in different DCs. When filtering, all DCs containing this data need to be opened (read into memory, decompressed, extracted), which incurs significant time and resource costs, resulting in poor job performance;

In the zipper step, both existing data closure and new data opening need to filter the table condition enddate='29991231 '. A table with n1 sharding has 2841 DCs for all data (approximately 180 million rows), but data with enddate='29991231' (approximately 800000 rows) is distributed across 2726 DCs, resulting in poor filtering performance of the smart index.

Inspection method

Through GNode Trace, it was found that the query performance is slow. In the Trace file, the Smart Scan section has a large number of found DCs, while the Scan result has a small number of rows. At this point, it can be determined that the filtered data is distributed discretely across multiple DCs to determine whether data organization and optimization are needed.

optimization method

Organize the table data every month and insert the data with enddate='29991231 'as a new dataset at the tail to reduce data distribution dispersion and delete the original data.

Reorganize table data every six months (Shrink Space) to release delete space.

Optimization operation steps

```
UPDATE p SET enddate=’29990000’ WHERE enddate=’29991231’;
INSERT p SELECT col1,…colx, ’29991231’ FROM p WHERE enddate=’29990000’ ;
DELETE FROM p_ 00ccard_ ambs WHERE enddate=’29990000’;
```

5.3.4 Optimize SQL statements

For massive amounts of data, the speed difference between low-quality SQL statements and high-quality SQL statements can reach hundreds or thousands of times. For databases, it is not just about simply implementing functions, but also meeting user performance requirements. SQL optimization can be achieved through the following methods to improve database performance indicators.

5.3.4.1 Storage optimization

GBase 8a MPP CLuster is a relational database based on column storage, which is different from traditional row storage databases. And in the subsequent improvement process, GBase 8a MPP Cluster currently supports two storage methods: column storage and row storage.

The following mainly introduces some characteristics of storage that bring value to optimization.

5.3.4.1.1 Inventory

In the column storage mode, the DML (Data Manipulation Language) operation of a column only scans the database page chain (column) corresponding to the column, which will not result in data access to the whole table, and can effectively reduce the I/O operations of the DML operation.

1. Optimization features:

- Smart index: Column storage databases generally adopt a sparse index technology, commonly known as smart index. The key to this index is to create an index (or statistical information) by dividing the data into blocks (pages), which can be used for coarse-grained filtering and filtering of data. This type of index takes up very little storage space, has low maintenance costs, and all maintenance work is done automatically without any manual intervention.

- Compression: Column based databases typically have higher compression ratios than row based databases due to storing data by column, feature similarity of data in the same column, and more targeted compression algorithms. Row based databases typically only have a compression ratio of 2-3 times, while column based databases can generally achieve a compression ratio of 5-10 times (even up to 20-40 times in some projects).
- Delay materialization: in the execution plan of the column database, no matter filtering, projection, connection, or aggregation operations, the column-oriented DBMS will not decompress the data until the final data is restored to the original data value. The benefits of doing so are: reducing CPU consumption, memory consumption, network transmission consumption, and ultimately storage requirements.

2. Optimization limitations:

- Scanning only for single column data has significant performance advantages. If it is in select multi column or select * mode, it can cause low I/O performance and seriously affect execution performance.
- Coarse grained indexes only have significant query performance for statistical information of DCs.

5.3.4.1.2 Mixed storage of rows and columns (row to column)

Due to the column storage architecture of GBase 8a MPP Cluster, when there are many columns and the accessed data records are very discrete, it can cause a large amount of discrete I/O, resulting in low I/O performance and effectiveness, seriously affecting execution performance.

The overall implementation logic of redundant row storage is to concatenate the data of each column that needs redundancy into a row and store it in a separate column, which is equivalent to adding a column of VARCHAR type to the original table, storing the corresponding row data. The management of row storage columns is generally the same as that of regular columns, but DCs are divided into multiple pages and can be loaded based on the page to be accessed without the need to read the entire DC.

The column column mixed storage function effectively improves I/O performance through redundant row storage, which can improve the performance of typical SELECT * From scenarios by an order of magnitude.

1. Optimization features:

- Support row and column compression storage to reduce redundancy.

- Rows and columns read data using smaller granularity Data Pages, reducing invalid I/O and improving query performance.
- Using gbase_ hybrid_ The store configuration parameter controls whether to use row storage data for comparison testing, troubleshooting errors, etc. The value of this parameter is:
  - 0: Not used;
  - 1: The server automatically determines whether to use it;
  - 2: Compulsory use, if available, must be used.

  Automatic judgment logic, used if the following conditions are met:
  - Fields are defined as rows and columns;
  - The value of RowsPerDC is less than or equal to the parameter_ gbase_ hybrid_ store_ The defined value of limit; RowsPerDC: refers to the number of records returned by the current query/the number of DCs hit.
- The storage redundancy method is flexible, and users can customize the fields in the row and column, mainly by selecting which fields should be included in the row and column based on the frequency of the projection column in the select statement.

2. Optimization limitations:

   The row storage column data is temporarily not used for scan, join, group, and other operations. Currently, it can only be used for the select (i.e. materialized) part with order by and related columns with order by in single table queries, such as:

```
SELECT * FROM t [ WHERE … ] ORDER BY ...;
```

3. Optimization parameters:
   - Whether to use row and column parameters when querying: gbase_ hybrid_ Store=0 (not used)/1 (automatic judgment/2 (mandatory use), this parameter defaults to 1.
   - Page size for row and column storage: gbase_ hybrid_ store_ page_ Size=<1k – 1G>, default value 32K.
   - The upper limit of the average number of DC returned records in the returned results:_ gbase_ hybrid_ store_ Limit=<1-65536>, default value of 100.
4. Other characteristics and limitations:
   - Support specifying rows and columns when creating tables;
   - Support adding rows and columns on existing tables;
   - Support deleting rows and columns;
   - Support the creation of multiple rows and columns on a table, but row and column names cannot be duplicate;

- The data updates of rows and columns are automatically maintained by the system and transparent to users. When the user executes Insert, Quick UPDATE, DELETE, Load, etc., the system will automatically update redundant data;
- Modifying the definition of row and column is not supported. You can only manually delete and then recreate the row and column;
- The original column definition of row storage columns does not allow deletion or modification of data attributes (but allows changing column names and order);
- Batch update operations are not allowed for the original column data (whole column replacement mode);
- The total length of all columns in the same grouped cannot exceed 32KB;
- Rows and columns support partition table, allowing indexes on the original columns.

5.3.4.1.3 Principles for Selecting Hash Distribution Columns

According to the characteristics of the data, for large table association and equivalent query conditions, it is necessary to consider building a hash distribution table. When selecting distribution keys, the data characteristics should be taken into account. The selection principles are as follows:

- Uniform distribution of data: Try to choose columns with high count (distinct) values as hash distribution columns to ensure uniform distribution of data;
- Distributed multi node operation: prioritize the JOIN between large tables, and try to make the JOIN condition columns of large tables hash distributed columns (this principle can also be referenced for JOIN related sub queries), so that JOIN between large tables can be directly distributed and published to each node for execution;
- Try to choose the frequent use of the 'grind by' column: Try to include the 'Hash' distribution column in the 'GROUP BY' column, so that the grouping aggregation can be completed in one step;
- Multi node operation: Select fields with high randomness in a certain data column to avoid hot queries on some nodes, resulting in uneven execution performance;

5.3.4.2 Fully Utilizing Hash Indexing

The use of indexes usually incurs maintenance costs, which can affect the performance

of data loading and DML operations. The actual use needs to be determined according to specific needs.

Hash Index can usually be used to improve the positioning efficiency of equivalent queries, especially for application scenarios where single table precise queries are the main focus, such as concurrent call order queries in telecommunications services (especially in scenarios where memory is basically sufficient).

In terms of usage, GBase 8a MPP Cluster must first perform intelligent index filtering. Afterwards, if a Hash Index is found to be established on the equivalent query condition column in the query criteria, then use the Hash Index. Otherwise, perform a full DC scan. This can be observed in the Trace Log for scenarios with real-time data loading. It is possible to first create an indexed temporary table to load data, and then insert the data from the temporary table into an indexed target table of the same structure or create an index on the temporary table. Disposing of index establishment in one go can significantly reduce the maintenance costs associated with indexing.

For detailed usage of Hash index, please refer to section 5.1.8.5.1 CREATE INDEX

be careful

- Indexing is a lossy optimization method, which usually incurs maintenance costs and affects the performance of data loading and DML operations. The actual use needs to be determined according to specific needs;
- Choosing columns to establish hash indexes should try to choose columns with fewer duplicate values, otherwise hash conflicts will be severe and affect the performance of hash indexes;
- Binary type columns are not suitable for using HASH indexes.

5.3.4.3 **Ordered data entry**

1. Performance advantages of orderly data warehousing:

- Improve the hit rate of intelligent index on DC
- Significantly improve query performance

2. Local range sorting

   In the use of databases, regular incremental data is stored, and before each batch of incremental data is stored, the batch data is sorted and then stored to ensure that the database data is organized within a local range and improve reading speed.

- **Example:**

Establish daily and monthly tables, sort daily table data before entering the monthly table, and sort monthly data at the end of the month.

a) Analyze SQL to identify the main query filter fields in the table (1 field);

b) Sort the data in the table according to the selected filter fields.

3. **Sort by:**

- External sorting: Use the sorting tool (PSort) to sort the data files, and then use the loading tool to load them into the table after sorting;
- Library sorting: Create a temporary table, store the unordered data in the temporary table, and then insert the data in the temporary table into the formal table by inserting into select *... order by XXX.

be careful

- After external sorting, it is still possible to cause data order disorder when loading into the warehouse, so it is recommended to use library sorting.

4. **Sort according to the scenario:**

- External sorting is suitable for non real-time loading businesses
- In library sorting is suitable for real-time loading of business

5.3.4.4 Semantic optimization

5.3.4.4.1 Sub query internal push optimization

GBase 8a MPP CLuster implements a sub query condition push optimization mechanism within the Express engine layer, which pushes the filtering conditions inward to filter out invalid data in temporary tables as early as possible, reduce the amount of data in temporary tables, and thereby improve SQL execution performance.

1. **Optimization features:**

- Conditional inference sub queries must meet the following requirements:
  - UNION sub queries, including UNION, UNION ALL, INTERSECT, and MINUS;
  - JOIN sub queries, including Left JOIN, Right JOIN, OUTER JOIN, INNER JOIN, FULL JOIN;
  - Single table sub query.
- The conditions for internal promotion need to meet:

- The filtering conditions of a single table can be inferred internally, and the expression must be a physical column, constant, constant expression, or NULL value;
- Multiple intra table filters that meet the filtering criteria of a single table connected through the AND logical operator;
- The constant true/false conditions can be inferred internally.

2. Optimization limitations:

- Filter condition constraints:
  - The conditional expression operator IN/NOT IN/BETWEEN AND is not supported;
  - Does not support the use of OR logical operators to concatenate internal conditions;
  - Single table filtering conditions do not support the use of functions, physical table expressions.
- Sub query constraints:
  - Subqueries with aggregate function, LIMIT clause, DISTINCT clause, OLAP function, HAVING, ORDER BY are not supported.

3. Optimization parameters:

Global parameters:_ gbase_ optimizer_ push_ Condition (a value of 0 indicates no optimization, a value of 1 indicates optimization, and the default value is 3).

5.3.4.4.2 Implicit surface materialization

Implicit tables represent tables for subqueries, such as SELECT COUNT (*) From (SELECT * From t1) tt;, Then tt is the implicit table. This optimization mainly focuses on not objectifying the projection parts that are not used in external queries in implicit tables. Simply put, it reduces useless objectified columns to improve SQL execution speed.

1. Optimization features:

Only external projection columns that are not used in the implicit table are not materialized.

For example, in the following example, b will not be materialized, for example:

```
SELECT tt.a FROM (SELECT a,b FROM t1) tt;
```

Optimization limitations:

- DISTINCT present in internal subquery

- UNION present in internal subquery
- There are RANK type OLAP functions (including RANK, DENSE-RANK, ROWNOUMBER, SUM() Over(), AVG() Over()) in internal subqueries. Even if these OLAP functions are not used externally, internal optimization will not be carried out, but it will not affect the optimization of other non OLAP function fields.

  Example:

```
SELECT a FROM (SELECT RANK()OVER(ORDER BY j) AS rank ,j AS a,k AS b,i AS c FROM t1) tt;
SELECT a FROM (SELECT RANK()OVER(ORDER BY j) AS rank ,j AS a FROM t1) tt;
```

5.3.4.4.3 OUTER JOIN to INNER JOIN

Optimization features:

- When the WHERE condition of table b in a Left JOIN b contains a non NULL judgment condition for a field, the Left JOIN can be converted to an INNER JOIN, thereby improving the performance of query optimization;
- When the WHERE condition of table a contains a non NULL judgment condition for a field in a RIGHTJOIN b, the RIGHTJOIN can be converted to INNER JOIN, thereby improving the performance of query optimization.

5.3.4.4.4 Rewrite IN to EXISTS (NOT IN=>NOT EXISTS)

GCluster 8a MPP Cluster automatically optimizes all statements with IN (subquery) into EXISTS (rewrited subquery) statements to improve performance.

1. **Optimization features:**

- Convert IN to EXISTS: Both single column and multi column IN subquery statements can be converted to EXISTS.

  Example:

  Original statement:

```
SELECT s1, s2 FROM t1 WHERE s2 IN (SELECT s1 FROM t1);
```

  After optimization:

```
<in_ optimizer>(t1.s2,<EXISTS>(select 1 AS Not_used FROM t1 WHERE (<cache>(t1.s2) = t1.s1)
```

- NOT IN to NOT EXISTS: Same as IN to EXISTS.

2. **Optimization limitations:**

- Statements with LIMITED/UNION (...)/OLAP/GROUP in IN subqueries cannot be optimized;
- Statements that support multi column In sub queries cannot be optimized if they have non column names such as (a, 10) IN (SELECT a, 10...).

3. **Optimization parameters:**

   _ gbase_ optimizer_ in_ Subselect controls whether to use optimization, which is enabled by default.

   In the configuration file _ gbase_ optimizer_ in_ subselect= 1

   Or Set in Client_ gbase_ optimizer_ in_ subselect= 1。

5.3.4.4.5 BETWEEN_ JOIN

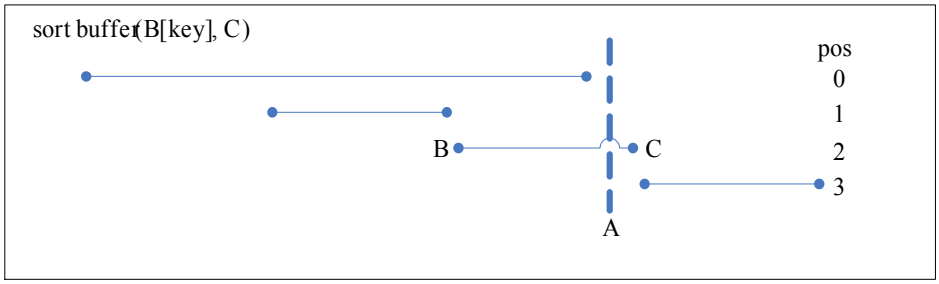
BETWEEN_ JOIN: Refers to a JOIN with connection conditions similar to "t1. A BETWEEN t2. B AND t2. C", including the notation "t1. A>t2. B AND t1. A<t2. C", abbreviated as "A BETWEEN B AND C".

The following is an explanation of t1. A BETWEEN t2. B AND t2. C.

1. **Optimization principle:**

   Understanding (B, C) as line segments, 'A BETWEEN B AND C' means finding all line segments that intersect with A.

   The processing logic of "A BETWEEN B AND C" in Figure -59 is shown in the following figure

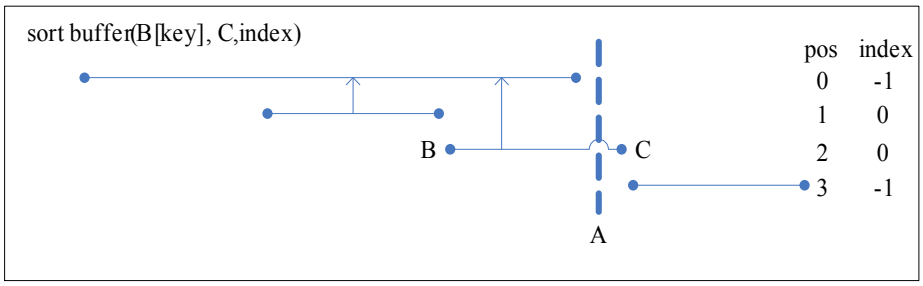


   You can first sort (B [key], C) as shown in the above figure, and then scan the t1 table. For any value A, use the binary method to find the corresponding position pos. Pos satisfies the requirement of "B (pos)<=A<B (pos+1)", as shown in the figure as 2. For this pos and all previous data, perform a check of "A between B and C", which requires checking (0, 1, 2).

   In order to improve the performance of the check, an index is added to the sort buffer, as shown in the following figure. The index points to the position of the previous interval that overlaps with the current interval (the up arrow in the figure). After finding the corresponding position pos, simply check the list composed of

indexes. For example, in the case shown in the figure, the corresponding A only needs to be checked (2,0).

The schematic diagram of Figure 510 Index is as follows:

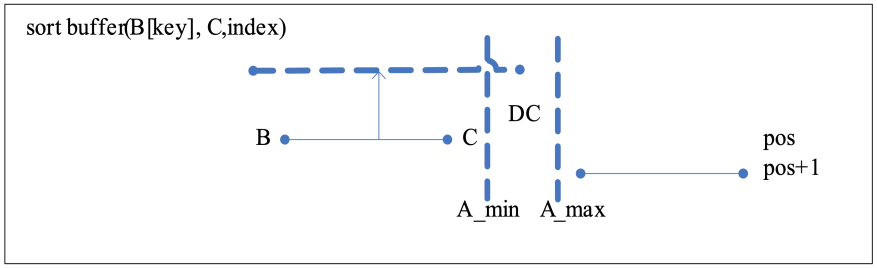


2. Optimization features:

The process of using intelligent indexes for filtering is shown in the following figure. If the shaded part does not intersect with all the intervals formed by (A, B), the current DC can be skipped, otherwise it cannot be skipped. The method is to perform the following checks on the current column A DC while scanning column A:

- Find A_ Position pos corresponding to min;
- If A_ If max<B (pos+1), continue; otherwise, the current DC cannot be skipped and ends;
- Along the index of pos for A_ Min for inspection, if no A is found to be met_ The line segment between B and C can skip the current DC, otherwise it cannot skip the current DC.

The schematic diagram of intelligent index filtering in Figure 511 is as follows:



When the interval segments composed of (B, C) are very sparse, and A has sequences, the intelligent index of column A will play an important role.

3. Optimization limitations:

Only when the t2 table is very small (relative to the t1 table), the optimization performance is very good. When the t2 table is large (such as reaching 1/2 of the t1 table), the performance is poor. It is recommended to follow the original logic and turn off this parameter.

4. Optimization parameters:

gbase_ optimizer_ between_ Join=0 (off)/1 (on default)/2 (on, and connections with equivalent conditions also prioritize using between joins).

5.3.4.4.6 Optimization of SUBSTR function

The cost of function calls for conditional columns is high, and smart indexes cannot be used. After converting to a like operation, on the one hand, smart indexes may be used, and on the other hand, SQL Function calls can be removed, which usually greatly improves performance.

1. **Optimization features:**

   Convert the character function in the filter criteria (currently only substr is supported) into a like operation to improve query performance.

   For example:

   Before optimization:

```
SELECT * FROM t1 WHERE SUBSTR(a, 2, 2)='ab';
```

After optimization

```
SELECT * FROM t1 WHERE a LIKE '_ ab%';
```

2. **Optimization limitations:**

- If there is no len parameter, it cannot be optimized: substr (a, 2)='ab 'cannot be optimized to a like'__ Ab% '(not equivalent);
- The len parameter and string length are not the same and cannot be optimized: substr (a, 2, 3)='ab 'cannot be optimized to a like'_ ab%';
- The first parameter of a substr must be a column name, otherwise it cannot be optimized: substr (a+1, 2, 3);
- If the column used by the substr is not a character column, it cannot be optimized;
- If the pos/len parameter of the substr is negative, it cannot be optimized;
- Currently, only the equal operation has been optimized, and other operations such as in are not currently optimized;
- If the right side of the equal sign is not a string, it cannot be optimized, such as substr (a, 2, 3)=substr (b, 2, 3).

3. **Optimization parameters:**

   Parameters_ gbase_ optimizer_ substr_ to_ Like to choose whether to use optimization: 0 (not used), 1 (default value, used).

5.3.4.4.7 OR optimization

If there is an OR condition in the WHERE part of OUTER JOIN, GBase 8a MPP Cluster will perform JOIN first and then filter the WHERE part during execution, which results in poor performance. To solve the performance issues of OR operation and "OUTER JOIN+OR", an OR optimization mechanism is introduced.

1. **Optimization features:**

   Organize a series of single table conditions containing OR into a composite condition, so that overall, this composite condition is no different from ordinary single table conditions, that is, from the outermost layer, it is equivalent to no OR condition, which also solves the performance problem of OUTER JOIN+OR.

   For example:

```
WHERE t1.a > 0 AND (t1.b > 0 OR t1.c > 0)
```

   The original execution logic will expand the conditions to only the outermost layer being an or condition, and all the inner layers being an and condition:

```
WHERE (t1.a > 0 AND t1.b > 0) OR(t1.a > 0 AND t1.c > 0)
```

   Under the new mechanism, the optimized logic is:

```
WHERE t1.a > 0 AND filter_ condtition(t1)
filter_ condtition(t1) := t1.b > 0 OR t1.c > 0
```

   The optimization prerequisite is that a series of conditions connected by OR can only be related to one table, so that the conditions can be optimized into a composite condition.

2. **Optimization limitations:**

   For OR conditions related to multiple tables, optimization is not possible because the vast majority of OR conditions in practical applications are only related to one table. If they are related to multiple tables, the most primitive execution logic can be used.

3. **Optimization parameters:**

Parameters_ gbase_ optimizer_ or_ The condition controls whether to use new OR optimization, which is enabled by default.

5.3.4.4.8 JOIN optimization

```
Null_ only： SELECT t1.*,t2.* FROM t1 LEFT JOIN t2 on (t1.a=t2.a) WHERE
t2.id IS NULL；
```

explain

This statement will first make a join before modification, and then filter the join results using the is null condition; After modification, the optimizer found that the is null condition is on the table filled with NULL values. When making a join, the matching results of the join are automatically ignored, and only the results filled with NULL are retained, thereby improving performance.

```
Distinct_ only：SELECT ts.a,ts.b FROM    tt1    ts , tt ta WHERE ta.a>ts.a GROUP
BY ts.a,ts.b；
```

explain

This statement was treated as a regular JOIN before modification, using line numbers to save the results, resulting in 4 records (two of which are duplicate). After modification, the optimizer found that it satisfies the distinct requirement_ Only optimization uses a filter to save the results during JOIN, and results with the same line number are automatically removed, resulting in only 2 records being generated.

1. Optimization features:

- Null_ Only optimization:

  In the original logic, if there is an is null condition on the outer dimension (table filled with NULL values) in OUTER JOIN, this condition will be set to Delay and executed after JOIN. But if the column itself of the is null operation does not have a NULL value, optimization can be carried out because the final result only contains the supplemented NULL value, that is, this NULL ONLY optimization: when doing this OUTER JOIN, all matching results are discarded, and only the supplemented NULL value in the JOIN is retained.

- Distinct_ Only optimization:

  This optimization is to identify JOINs whose final result can only be distinct values, and use filters to save the JOIN results. The conditions that need to be met to use this optimization:

  - The projection column only involves one table;
  - No HAVING;
  - There is a GROUP BY but no aggregation;
  - No GROUP BY, no aggregation, but there is DISTINCT in the projection

column;

- No GROUP BY has aggregates, but the aggregate function is MIN/MAX.

2. **Optimization limitations:**

Full join does not support null_ Only optimization.

5.3.4.4.9 HASH JOIN optimization

In analytical databases that require large-scale data processing, the join process accounts for a significant proportion of query time, with HASH JOIN being the most common JOIN type. Therefore, improving the performance of the HASH JOIN operator is of great significance for improving overall query performance.

The One Pass Hash Join algorithm is an optimization algorithm for HASH JOIN. By hashing both JOIN parties first, performing HASH JOIN on the same component slice, and finally merging the JOIN results of each group, the JOIN time consumption shows a linear growth trend with the increase of data volume. This improvement improves the adaptability of the HASH JOIN algorithm to data growth, while also significantly improving query performance in large data volumes.

- Enabling One Pass Hash Join requires configuring GNode configuration file options_ gbase_ one_ pass_ hash_ join;
- When_ gbase_ one_ pass_ hash_ When join=0 or not set, use the GBase 8a MPP CLuster original Hash Join algorithm;
- When_ gbase_ one_ pass_ hash_ When join=1, automatically evaluate whether to use the One Pass Hash Join algorithm based on the amount of data in the traverse table and the size of the hash buffer. If the hash buffer cannot accommodate the hash table created by the entire traverse table, then use the One Pass Hash Join algorithm; otherwise, use the original Hash Join algorithm;
- When_ gbase_ one_ pass_ hash_ When join=2, use the One Pass Hash Join algorithm without making any judgments.

be careful

The current One Pass Hash Join implementation only takes effect when in parallel, which requires setting gbase in the configuration file_ parallel_ execution = 1。

5.3.4.4.10 SELECT INTO SERVER optimization

SELECT INTO SERVER is used to achieve data transmission between GBase 8a MPP Cluster servers. It can redistribute the data in the table across various nodes of the cluster when needed to achieve certain specific operations. When there are multiple target nodes, a hash method can be set to send data groups. And each computing task can be executed in parallel.

- **Parameter configuration**

  Adding to a standalone configuration file_ gbase_ use_ new_ Sis parameter:

  - _ gbase_ use_ new_ Sis=0 (default), indicating that the original method is used and optimization does not work;
  - _ gbase_ use_ new_ Sis=1, using the optimized select into server.

5.3.5 Other optimization suggestions

5.3.5.1 Try not to use cursors as much as possible

The operation of a cursor is similar to taking out the values of each row and performing a series of processing. If the cursor can be removed and replaced with a SQL statement containing multiple related sub queries, the performance will be greatly improved.

5.3.5.2 Try to use VARCHAR without CHAR

- The blank space of CHAR may affect performance;
- The association between CHAR and VARCHAR can lead to incorrect associations.

5.3.5.3 Try to use UNION ALL as much as possible without UNION

Due to the need for one deduplication operation, deduplication has a significant impact on performance. Try to ensure that the same data is only stored once and there is no duplicate data between different tables. Performing UNION ALL will greatly improve performance.

5.3.5.4 Avoiding direct returns of oversized result sets

For query result sets that exceed 10000, especially those with millions or tens of millions, direct return of the result set should be avoided. The original select should be modified to insert select, which means inserting the query results into a result table or adding the - q parameter when outputting from the client.

5.3.5.5 High precision DECIMAL may slow down performance

If low precision decimals are used before the system upgrade, the upgraded high precision decimals may slow down performance due to the association of high precision decimals, which require more resources for operations such as value taking. However, this slowdown in performance is normal, and as long as it is within a reasonable and acceptable range, this issue does not need to be considered.

5.3.5.6 INSERT INTO ... SELECT ... GROUP BY serial

- **Phenomenon**: Insert INTO SELECT ... GROUP BY ... Parallelism is divided into HASH and involves multiple aggregations, resulting in the serial execution of the Insert part.
- **Cause**: The data is divided according to HASH. When the execution cannot be completed in a single run, the GROUP BY operation occupies the thread. As a result, there is no idle thread in the thread pool, and the INSERT can only be performed serially.
- **terms of settlement**:
  - If the number of machine cores is large (>=32), the parallelism can be reduced to half of the number of cores, and the thread pool can use the default value (number of cores);
  - If the number of cores is less than 32, the parallelism can be increased to twice the number of cores.

5.3.5.7 **Precautions for hash distribution columns**

- During the use of hash column fields, it is prohibited to add functions such as LTRIM to handle operations. This will damage the hash distribution and must be removed to ensure the correctness of field data externally.
  For example, the RTRIM and LTRIM added to field col1 in the GROUP BY and Insert INTO SELECT projection columns break the hash distribution and must be removed.
- If the GROUP BY statement contains a hash column, place the hash column first.
- If there are multiple JOIN columns with a hash column JOIN, place the hash column JOIN at the top.

5.3.5.8 **Scan does not support parallel scenarios**

Firstly, when the number of data rows involved<=parameter gbase_ parallel_ At threshold, parallelism is not supported.

1. The data of a scan must exceed one DC in order to be parallelized, as the scan is segmented by DC.
   It needs to be explained here that it is not necessary to have a select count (*) exceeding 65536, but rather to have more than one DC. For example, if there are two DCs, each with 65535 deleted entries, the value of select count (*) is 2, but in reality, there are still two DCs, so it can be done in parallel.
2. Hierarchical query pseudo column level does not support parallelism.
   For example:

```
DROP TABLE IF EXISTS t1;
CREATE TABLE t1(i INT);
INSERT INTO t1 VALUES(0),(1),(2);
SELECT i,level FROM t1 START WITH i = 0 CONNECT BY prior i+1 = i;
SELECT i,level FROM t1 START WITH i = 0 CONNECT BY prior i+1 = i
GROUP BY i,level HAVING LEVEl > 0;
```

3. Some functions do not support:
   1) RAND function does not support parallelism without parameters or with constant parameters

```
SELECT ... FROM t1 WHERE RAND() > 0;
SELECT ... FROM t1 WHERE RAND(100) > 0;
```

2） REGEXP (parameter existence field not supported, all constants supported)

```
SELECT ... FROM t1 WHERE a REGEXP b;
```

3） :=Operation (parameter existence field not supported, all constants supported)

```
SELECT ... FROM t1 WHERE @cnt := i > 0;
```

4. Custom functions created with SQL do not support parallelism, UDF supports:

For example:

```
CREATE FUNCTION f() RETURNS INT
BEGIN
    ...
END
```

This is not supported.

5. Full text indexing does not support parallelism.

5.3.6 Database performance monitoring

5.3.6.1 Cluster monitoring tools

The GBase 8a MPP Cluster monitoring tool is one of the components of the GBase 8a MPP Cluster developed by Nanda General Data Technology Co., Ltd. Provide monitoring data, timely alarm function, intuitive trend display, reliable data distribution view, and database connection thread status display.

The GBase 8a MPP Cluster monitoring tool mainly monitors the running status, resource utilization, network communication status, and other information of the cluster node servers in the GBase 8a MPP Cluster deployment environment. It can provide reliable basis for users to monitor the operation of the cluster and its cluster points.

- Provide monitoring data;
- Timely alarm function;
- Intuitive trend display;
- Reliable data distribution view;
- Status display of database connection threads

Figure -512 Visual monitoring tool

5.3.6.2 **Log View**

GBase 8a MPP CLuster provides a rich variety of logs with different purposes, including:

- Trace log (records the complete execution process of SQL, mainly used for analyzing performance).
- System log (also known as error log, which records important operations such as database service startup and shutdown, and can also record program stacks for abnormal situations such as database service downtime, which can assist developers in troubleshooting).
- Express log (records important information during the internal execution process of the express engine, including exceptions, and sometimes can be used for error checking).
- SQL log (also known as general log, which can be used to record SQL statements executed by the database)
- Slow logs (can be used to find slow statements).

  Through various logs, we can help troubleshoot errors and analyze performance. For the use of most logs, please refer to product manuals and other documents. Below, we will focus on using logs for troubleshooting and performance analysis.

5.3.6.2.1 Execution Log

Express.log records warnings and errors during SQL execution.

Set gcluster in the gcluster configuration file_ log_ After level=15, the complete execution plan can be output to the express.log file. By default, only warnings and errors that occur during the execution process will be output to the express.log file.

- **Common errors:**
  - Got error 28 from storage engine

    The temporary space is full. Please configure the parameters for the temporary space (default is in the/tmp directory):

```
tmpdir = /opt/tmp
```

```
2012-07-19 02:57:30.355 [ERROR] <CreateTempTable|14825>: Create table fail with SQL: CREATE
TABLE `gctmpdb`._tmp_n1_t23_1_513743_0x40e542a0_0x446e9b8(`date '2011-12-04' + interval '$i'
day     ` VARCHAR(29)) ENGINE = express nolock
CAUSE: Incorrect column name 'date '2011-12-04' + interval '$i' day     '
```

  - Port number 6002 is occupied
  - Create temporary table fail

5.3.6.2.2 Audit logs

Audit logs are used to monitor database operations performed by users, and the main contents recorded include:

- User login methods (CAPI, ODBC, JDBC, ADO)
- Return the result set (number of rows, execution time)
- Logged in user and IP
- Start execution time
- Executed SQL statements

$GCLUSTER_ Audit under BASE/log/gcluster_ log

```
# Time: 140115 17:21:55
# User@Host: root[root] @ localhost []
# Query_time: 0.011142 Rows: 0
# SET timestamp=1389777715;
# sql_text: select * from hj h1 where h1.name in(select name from hj);
# Connect Type: CAPI;
```

- Additionally, the following configuration method can be used to store audit logs in system tables:

  Global level variable:

SET GLOBAL log_ output = 'table';

- Example: Viewing audit logs using system tables

```
gbase> SELECT start_ time,user_ host,query_ time,rows,LEFT(sql_text, 30),
conn_ type FROM gbase.audit_ log;
+---------------------+---------------------------------+
| start_ time         | user_ host                      |
+---------------------+---------------------------------+
| 2013-10-09 17:21:08 | root[root] @ localhost []       |
| 2013-10-09 17:21:22 | root[root] @    [192.168.10.116]  |
| 2013-10-09 17:21:22 | root[root] @ localhost []       |
| 2013-10-09 17:21:32 | gbase[gbase] @    [192.168.10.116] |
| 2013-10-09 17:21:32 | root[root] @ localhost []       |
| 2013-10-09 17:21:32 | root[root] @ localhost []       |
| 2013-10-09 17:21:45 | root[root] @ localhost []       |
| 2013-10-09 17:21:52 | root[root] @ localhost []       |
| 2013-10-09 17:21:58 | root[root] @ localhost []       |
| 2013-10-09 17:22:05 | root[root] @ localhost []       |
| 2013-10-09 17:22:10 | gbase[gbase] @    [192.168.10.116] |
| 2013-10-09 17:22:10 | root[root] @ localhost []       |
| 2013-10-09 17:22:17 | root[root] @ localhost []       |
+---------------------+---------------------------------+

+-----------------+------+------------------------------+----------+
| query_ time     | rows | LEFT(sql_text, 30)           | conn_ type|
+-----------------+------+------------------------------+----------+
| 00:00:00.006397 |    0 | SET GLOBAL log_ output = 'table| CAPI     |
| 00:00:00.000282 |    0 | Connect                      | CAPI     |
| 00:00:00.025018 |    0 | DROP USER tzt                | CAPI     |
| 00:00:00.000054 |    0 | Connect                      | CAPI     |
| 00:00:00.000175 |    0 | DROP DATABASE test           | CAPI     |
| 00:00:00.111946 |    1 | SELECT DATABASE()            | CAPI     |
| 00:00:00.000086 |    0 | CREATE USER tzt identified by | CAPI    |
| 00:00:00.439480 |    0 | GRANT ALL ON *.* TO tzt@'%'  | CAPI     |
| 00:00:00.000387 |    0 | CREATE DATABASE test         | CAPI     |
| 00:00:00.000025 |    0 | USE test                     | CAPI     |
| 00:00:00.000384 |    0 | Connect                      | CAPI     |
| 00:00:00.000144 |    0 | CREATE TABLE t1(i int)       | CAPI     |
| 00:00:00.004527 |    2 | INSERT INTO t1 VALUES (1),(2) | CAPI    |
+-----------------+------+------------------------------+----------+
13 rows in set
```

Note: If gccli connects to the cluster without the h parameter, it defaults to using UDS (unix domain socket) connection instead of IP and PORT. Therefore, when recording audit logs, the user_ Host and host_ The IP record in the IP is empty.

5.3.6.2.3 Trace Log

The TRACE of GBase 8a MPP Cluster is used to view the execution plan of SELECT statements and analyze their performance bottlenecks. The system tables related to this include SQL_ Trace (TRACE information of the SELECT statement) and audit_ Log (SQL history information of execution).

- The general execution process for an SQL statement is:

  smart scan -> scan -> join -> aggregation -> sort -> materialization -> send result

  The trace record is very long, and finding the step with the longest execution time between two steps is the problem to be located.

```
2013-08-30 11:15:55.584 [M:  18K,   0B,D:   0B] [DC:    0,    0] SQL Statement:
select * from test.dctest where pwd=111111
2013-08-30 11:15:55.584 [M:  18K,   0B,D:   0B] [DC:    0,    0] Start Query Execution
2013-08-30 11:15:55.584 [M:  18K,   0B,D:   0B] [DC:    0,    0] Tables:
2013-08-30 11:15:55.584 [M:  18K,   0B,D:   0B] [DC:    0,    0] T0:         dctest(test.dctest),    24 rows,       1 DC
2013-08-30 11:15:55.584 [M:  18K,   0B,D:   0B] [DC:    0,    0] Condition Weight (non-join):
2013-08-30 11:15:55.584 [M:  18K,   0B,D:   0B] [DC:    0,    0] cnd(0):      dctest.pwd BET. 111111 AND 111111       (0)
2013-08-30 11:15:55.584 [M:  18K,   0B,D:   0B] [DC:    0,    0] BEGIN Smart Scan
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:    0,    0] T0:         total 1 DC,     found 1 DC to scan(with 0 FULL DC).
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:    0,    0] BEGIN Scan
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:    1,    0] cnd(0):      scanned 24 rows,        found 5 rows
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:    1,    0] T0:         total 1 DC,     found 1 DC after scan(with 0 FULL DC
).
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:    1,    0] BEGIN Join
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:    1,    0] Join done
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:    1,    0]
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:    1,    0] BEGIN Materialization(5 rows)
2013-08-30 11:15:55.608 [M:   2M,   0B,D:   0B] [DC:    6,    0] Materialization 0 to 4 already
2013-08-30 11:15:55.608 [M:   2M,   0B,D:   0B] [DC:    6,    0] Serially materialization 0 to 4 already
2013-08-30 11:15:55.608 [M:   2M,   0B,D:   0B] [DC:   11,    0] Send 5 rows already
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:   11,    0] Serially Materialization Done.
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:   11,    0] ResultSender: send 5 rows.
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:   11,    0] output result done.

2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:   11,    0] SUMMARY
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:   11,    0] elapsed time:              00:00:00.024
2013-08-30 11:15:55.608 [M:  18K,   0B,D:   0B] [DC:   11,    0] data loaded from storage:      0B,  0s,     0 DC.
2013-08-30 11:15:55.609 [M:  18K,   0B,D:   0B] [DC:   11,    0] data decompressed:             0B,  0s.
2013-08-30 11:15:55.609 [M:  18K,   0B,D:   0B] [DC:   11,    0] temp space IO stats:
2013-08-30 11:15:55.609 [M:  18K,   0B,D:   0B] [DC:   11,    0] CB   write(   0B, 0time, 0sec),       read(   0B, 0time, 0sec)
2013-08-30 11:15:55.609 [M:  18K,   0B,D:   0B] [DC:   11,    0] SRT  write(   0B, 0time, 0sec),       read(   0B, 0time, 0sec)
2013-08-30 11:15:55.609 [M:  18K,   0B,D:   0B] [DC:   11,    0] GDC  write(   0B, 0time, 0sec),       read(   0B, 0time, 0sec)
2013-08-30 11:15:55.609 [M:  18K,   0B,D:   0B] [DC:   11,    0] MAT  write(   0B, 0time, 0sec),       read(   0B, 0time, 0sec)
2013-08-30 11:15:55.609 [M:  18K,   0B,D:   0B] [DC:   11,    0] ================================================
```

Common settings for trace

The most commonly used performance analysis method currently is through a more detailed trace analysis.

- The relevant parameters involved include:
  - Trace log switch: gbase_ sql_ Trace is used to control whether to open trace logs.
  - Trace log level: gbase_ sql_ trace_ Level is used to control the level of detail in logs, divided into several different levels. When analyzing performance, it is recommended to set it to 7 or 15, which can output the most detailed logs.

5.4 Stored Procedures and Functions

5.4.1 summary

A stored procedure is a set of SQL statements that can perform specific functions and are compiled and stored in a database. When executing a stored procedure, users need to specify the name of the stored procedure and provide parameters (if the stored procedure contains parameters).

Stored procedures are very useful in the following situations:

- When multiple client applications are written in different languages or run on different platforms, but need to perform the same database operations;
- When safety is very important. For example, banks use stored procedures for all commonly used operations. This provides a consistent and secure environment, and stored procedures can ensure that every operation is correctly written to the log. With this setting, applications and users will not be able to directly access the data table and can only execute specific stored procedures;
- Stored procedures can improve performance because less information needs to be passed between the server and the client. The negative impact is that it increases the burden on the database server, as more tasks are executed on the server side and fewer tasks need to be executed on the client side (application).
- Stored procedures allow users to use function libraries in the database server. This is precisely the characteristic of modern application programming languages, such as programming through the use of classes. These client application language features, whether or not applied to database side design, are still very beneficial for programmers to adopt this approach.

5.4.2 Create stored procedures/functions

Function Description

Stored procedures and functions are programs created by the CREATE PROCESS and CREATE JUNCTION statements.

- The stored procedure is called through a CALL statement, and the return value can only be obtained by outputting variables. Functions can be called from within a statement like other functions (by calling the function name) and return a scalar value. Stored programs (procedures and functions) can also call other stored programs (procedures and functions);
- Each stored procedure or function is associated with a specific database. When stored programs (procedures and functions) are called, the implicit USE *database_ The name* is executed (completed when the stored program (procedure and function) ends), and the USE statement is not allowed in the stored program (procedure and function). Users can use database names to qualify stored program (procedure and function) names. This can be used to indicate stored programs (procedures and functions) that are not in the current database. For example, to call a stored procedure p or function f associated with a gbase database, the user can use CALL gbase. p() or gbase. f(). When a database is deleted, all related stored programs (procedures and functions) are also deleted;
- GBase 8a MPP Cluster can use standard SELECT statements in stored procedures. In this way, the results of a query are simply and directly transmitted to the client. Multiple SELECT statements generate multiple result sets, so the client must use a GBase 8a MPP Cluster client library that supports multiple result sets;
- To create a stored program (procedure and function), you must have the CREATE ROUTINE permission. If the user creates a PROCESS | JUNCTION, it will automatically grant the user the ALT ROUTINE and EXECUTE permissions for that PROCESS | JUNCTION. If the update log is enabled, users may need SUPER permission;
- By default, stored programs (procedures and functions) are associated with the current database. To explicitly associate a procedure with a database, users need to write its name in VC format when creating stored programs (procedures and functions)_ name.database_ name.sp_ name;
- There must be a parameter list in parentheses. If there are no parameters, an empty parameter list should be used (); The default parameter is the IN parameter. If you want to specify a parameter as a different type, please specify the parameter type before the parameter name;

- When using the RETURNS clause (which can only be specified with a ACT ION clause) to indicate the return type of a function, the function body must contain a RETURN statement;
- If a stored procedure or function obtains the same results for the same input parameters, it is considered 'Deterministic', otherwise it is' Not Deterministic '. The default is NOT DERMINISTIC;
- In terms of replication, using the NOW () function (or its synonym) or RAN D () does not generate a non deterministic program. For NOW (), the update log includes a timestamp and can be replicated correctly. If called only onc e in a program, RAND() can also be copied correctly;
- The current DERMINISTIC feature is acceptable, but it is not used by the o ptimizer. However, if the update log is activated, this feature will affect whet her GBase 8a MPP Cluster accepts the definition of the process;
- The following characteristic parameters provide data usage information for the program:

  1. The SQL Security parameter is used to indicate whether the execution per mission of this program is granted to the creator or caller. The default value is DEFINER. Creators and callers must have access to the database related to the program. To execute stored programs (procedures and functions), it is necessary to have EXECUTE permission. The user who must have this permi ssion is either the definer or the caller, which depends on how to set the S QL Security feature;

  2. The COMMENT statement is an extension of GBase 8a MPP Cluster and can be used to describe stored procedures. You can use the SHOW CREAT E PROCESS, SHOW CREATE JUNCTION statements to display this inform ation.

- GBase 8a MPP Cluster allows stored programs (procedures and functions) to contain DDL statements (such as CREATE and DROP) and SQL transaction statements (such as COMMIT). This is not required by the standard, it is onl y a specific implementation;
- The statement that returns the result set cannot be used in a function. These statements include SELECT statements that do not use INTO to assign colu mn values to variables, SHOW statements, and so on. A statement that retur

ns a result set at function definition returns a 'Not allowed to return a result set from a function' error (ER_SP_NO-RETSET_IN_FUNC). For statements that only return a result set while the function is running, an error (ER_SP_BADSELECT) of 'PROCESS% s can't return a result set in the given context' is returned.

Grammar format

```
stored procedure
CREATE PROCEDURE <proc_ name>([<proc_parameter_1>[,…]
[,proc_parameter_n]])
[characteristic ...] routine_ body

CREATE FUNCTION <func_ name>([<func_parameter_1>[,…]
[,func_parameter_n]])
RETURNS    type
[characteristic ...] routine_ body

Proc_ parameter:
{IN | OUT | INOUT } param_ name type

Func_ parameter:
Param_ name type

Characteristic:
   LANGUAGE SQL
   | [NOT] DETERMINISTIC
   | { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL
DATA }
   | SQL SECURITY { DEFINER | INVOKER }
   | COMMENT 'string'
```

Table 5167 Parameter Description

Field Name	Explanation of Meaning
proc_ name	The name of the stored procedure to be created must be unique under the same VC. Stored procedure names only allow a-z, A-Z, 0-9, and underscores, and cannot contain only numbers.

Field Name	Explanation of Meaning
proc_ parameter	Define the parameters of the stored procedure, and the definition format for each parameter is:<parameter direction><parameter name><parameter data type>.
func_ name	The name of the function to be created must be unique under the same VC. Function names only allow a-z, A-Z, 0-9, and underscores, and cannot contain only numbers.
func_ parameter	Define the parameters of a function, with each parameter defined in the following format:<parameter name><parameter data type>.
IN | OUT | INOUT	To determine whether a parameter is input, output, or input-output, only one of IN, OUT, or INOUT can be taken.
param_ name	Parameter names must be unique within the same stored procedure/function, only a-z, A-Z, 0-9, and underscores are allowed, and cannot only contain numbers;
type	Parameter data type, with values of data types supported by GBase 8a MPP Cluster
LANGUAGE SQL	Explanation routing_ The body section is composed of SQL statements, which are the unique values of the LANGUAGE feature, indicating that only SQL statements are currently supported.
[NOT] DETERMINISTIC	Indicates whether the result of the stored procedure execution is deterministic. Deterministic indicates that the result is certain. Every time a stored procedure is executed, the same input will result in the same output. Not Deterministic indicates that the result is uncertain, and the same input may result in different outputs. If no value is specified, the default is NOT DEFORMINISTIC
CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA	Represents the restrictions on the use of SQL statements by subroutines. CONTAINS SQL indicates that the subroutine contains SQL statements, but does not contain statements that read or write data. By default, the system specifies it as CONTAINS SQL; NO SQL indicates that the subroutine does not contain SQL statements; READS SQL DATA: Indicates that the subroutine contains statements for reading data; MODIFIES SQL DATA indicates that the subroutine contains statements for writing data.
SQL SECURITY { DEFINER | INVOKER }	Indicate who has permission to execute. DEFINER indicates that only the definer can execute, and by default, the system specifies DEFINER; INVOKER indicates that a caller with permission can execute.

Field Name	Explanation of Meaning
COMMENT 'string'	Annotation information, which can be used to describe stored procedures or functions
routine_ body	It is a combination of a series of SQL statements that contain some data operations to complete certain functional logic. You can use BEGIN END to represent the beginning and end of SQL code

explain

- When defining a stored procedure, the parentheses after the name of the stored procedure/function are required, and cannot be omitted even if there are no parameters;
- If the route in the stored procedure or function_ If the body only contains one SQL statement, BEGIN and END can be omitted. Otherwise, BEGIN must be used when defining stored procedures/functions The END structure organizes related SQL statements together to form a route_ body;
- Stored procedures and functions can be nested.

Example

The following is an example of a simple stored procedure that uses IN and OUT parameters. This example uses the delimiter command to change the statement d elimiter from ";" to "//" before defining the stored procedure. This allows the ';' delimiter used in the stored program body to be passed to the server instead of b eing interpreted.

be careful

- The purpose of the 'DELIMITER//' statement is to set the SQL terminator to '//', as the default statement terminator is a semicolon ';'. To avoid conflicts with the SQL statement terminator in a stored procedure, it is necessary to use DELIMITER to change the terminator of the stored procedure and end the stored procedure with 'END//'.
- After the stored procedure is defined, use 'DELIMITER;' to restore the default terminator. DELIMITER can also specify other symbols as terminators.
- When writing, please note that there are spaces before the closing character. For example, there should be a space before the end of statements such as "DELIMITER//" and "END//", and there should also be a space before the end of SQL statements.

Example 1: Creating a process_ Count stored procedure and call.

```
gbase> DELIMITER //
gbase> CREATE PROCEDURE proc_ count (OUT param1 INT,IN param2
VARCHAR(10))
      BEGIN
      SELECT COUNT(*) INTO param1 FROM ssbm.customer WHERE c_
nation= param2;
      END //
Query OK, 0 rows affected

gbase> CALL proc_ count(@count1,'JORDAN') //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> SELECT @count1;
+---------+
| @count1 |
+---------+
|      1182 |
+---------+
1 row in set
```

 **explain**

When using the delimiter command, users should avoid using the backslash bar (representing escape character in GBase 8a MPP Cluster).

Example 2: Create a hello function with parameters, use SQL functions to perform operations and return results.

```
gbase> DELIMITER //
gbase> CREATE FUNCTION hello (s CHAR(20)) RETURNS CHAR(50)
      RETURN CONCAT('Hello, ',s,'!'); //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> SET @result = hello('world');
Query OK, 0 rows affected

gbase> SELECT @result;
+----------------------------------------------------+
| @result                                            |
+----------------------------------------------------+
| Hello, world                    !                  |
+----------------------------------------------------+
```

```
1 row in set
```

explain

If the RETURN statement of a function returns a value of a different type than the value specified in the RETURNS clause, the returned value is cast to the return value type defined by the function.

Example 3: Creating fn_ The count function contains SQL statements in the procedure definition.

```
gbase> DELIMITER //
gbase> CREATE FUNCTION fn_ count (param   varchar(10)) RETURNS INT
        BEGIN
        SELECT COUNT(*)/5 INTO @count FROM ssbm.customer WHERE c_ nation= param;
        RETURN @count;
        END //
Query OK, 0 rows affected

gbase> DELIMITER ;

gbase> SET @result = fn_ count('JORDAN');
Query OK, 0 rows affected

gbase> SELECT @result;
+---------+
| @result |
+---------+
|       236 |
+---------+
1 row in set
```

5.4.3 Modifying Stored Procedures/Functions

Function Description

Used to change the characteristics of a stored procedure or function. The user needs the Alter ROUTINE permission to use this statement, which will automatically be granted to the creator of the subroutine.

Grammar format

```
ALTER {PROCEDURE | FUNCTION} <sp_ name> [characteristic ...]
characteristic:
{ CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }
| SQL SECURITY { DEFINER | INVOKER }
| COMMENT 'string'
```

Table 5168 Parameter Description

Parameter Name	Description
Sp_ name	The name of the stored procedure or function to be modified.
CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA	Represents the restrictions on the use of SQL statements by subroutines. CONTAINS SQL indicates that the subroutine contains SQL statements, but does not contain statements that read or write data. By default, the system specifies it as CONTAINS SQL; NO SQL indicates that the subroutine does not contain SQL statements; READS SQL DATA: Indicates that the subroutine contains statements for reading data; MODIFIES SQL DATA indicates that the subroutine contains statements for writing data.
SQL SECURITY { DEFINER | INVOKER }	Indicate who has permission to execute. DEFINER indicates that only the definer can execute, and by default, the system specifies DEFINER; INVOKER indicates that a caller with permission can execute.
COMMENT ‘string’	Annotation information, which can be used to describe stored procedures or functions

Example

Example 1: Modifying annotation information for stored procedures.

```
gbase> ALTER PROCEDURE proc_ count COMMENT 'new comment';
Query OK, 0 rows affected
```

Example 2: Modifying annotation information for a function.

```
gbase> ALTER FUNCTION fn_ count COMMENT 'new comment';
Query OK, 0 rows affected
```

5.4.4 Delete stored procedures/functions

Function Description

Used to delete a stored procedure or function. The user needs the Alter ROUTIN E permission to use this statement, which will automatically be granted to the cre ator of the subroutine.

Grammar format

```
DROP {PROCEDURE | FUNCTION} [IF EXISTS] <sp_ name>;
```

Table -5169 Parameter Description

Parameter Name	Description
IF EXISTS	If a stored procedure or function does not exist, it will prevent errors from occurring. Generate a warning that can be viewed using SHOW WARNINGS.
sp_ name	The name of the stored procedure or function to be deleted.

Example

Example 1: Deleting a stored procedure.

```
gbase> DROP PROCEDURE IF EXISTS proc_ count;
Query OK, 0 rows affected
```

Example 2: Deleting a function.

```
gbase> DROP FUNCTION IF EXISTS fn_ count;
Query OK, 0 rows affected
```

5.4.5 Calling stored procedures/functions

Function Description

Use the CALL statement to call a stored procedure that has already been created. Call a function using a SELECT statement.

Grammar format

```
CALL [database_name.]proc_ name([parameter_1[,...,parameter_n]]);

SET @Variable_ name = func_ name([parameter_1[,…,parameter_n]]);
```

```
SELECT @Variable_ name;
```

Table 5170 Parameter Description

Parameter Name	Description
proc_ name	The name of the stored procedure to be called.
parameter	Call parameters. If the stored procedure has parameters, they must be assigned values in the order and type specified in the stored procedure definition, and for OUT and INOUT parameters, the OUT and INOUT keywords must be specified.
func_ name	The name of the function to be called.
Variable_ name	Variable name, which can assign the execution result of a function to a variable.

explain

- For parameterless stored procedures, do you want to add parentheses after the<stored procedure name>to ensure consistent execution results.
- GBase 8a MPP Cluster uses a SELECT statement to view the execution results of the calling function.
-

Example

Example 1: Example of calling a stored procedure.

```
gbase> USE test;
Query OK, 0 rows affected

gbase> DELIMITER //
gbase> DROP PROCEDURE proc_ count //
Query OK, 0 rows affected

gbase> CREATE PROCEDURE proc_ count (OUT param1 INT,IN param2
varchar(10))
            BEGIN
            SELECT COUNT(*) INTO param1 FROM ssbm.customer
      WHERE c_ nation= param2;
            END //
Query OK, 0 rows affected

gbase> CALL proc_ count(@count1, 'JORDAN') //
Query OK, 0 rows affected

gbase> DELIMITER ;
```

```
gbase> SELECT @count1 ;
    +---------+
    | @count1 |
    +---------+
    |    1182 |
    +---------+
    1 row in set
```

Example 2: Example of calling a function.

```
gbase> USE test;
Query OK, 0 rows affected

gbase> DELIMITER //
gbase> DROP FUNCTION hello //
Query OK, 0 rows affected

gbase> CREATE FUNCTION hello (s CHAR(20)) RETURNS
VARCHAR(50)
            RETURN CONCAT('Hello, ',s,'!') //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> SET @result = hello('world');
Query OK, 0 rows affected

gbase> SELECT @result;
    +-----------------------------+
    | @result                     |
    +-----------------------------+
    | Hello, world                ! |
    +-----------------------------+
    1 row in set
```

5.4.6 Viewing the Status of Stored Procedures/Functions

5.4.6.1 Viewing Stored Procedure/Function Creation Statements

Function Description

View the creation of a given stored procedure or function.

Grammar format

```
SHOW CREATE {PROCEDURE | FUNCTION} <sp_ name>;
```

Example

Example 1: Displaying the Create Stored Procedure proc_ 1 statement.

```
gbase> SHOW CREATE PROCEDURE vc1.demo.proc_ 1\G
*************************** 1. row ***************************
                Procedure: proc_ one
                  sql_ mode: PIPES_ AS_ CONCAT,ANSI_ QUOTES,IGNORE_
SPACE,ONLY_ FULL_ GROUP_ BY,NO_ AUTO_ VALUE_ ON_
ZERO,STRICT_ ALL_ TABLES,NO_ ZERO_ IN_ DATE,NO_ ZERO_
DATE,NO_ AUTO_ CREATE_ USER,NO_ ENGINE_ SUBSTITUTION,PAD_
CHAR_ TO_ FULL_ LENGTH
      Create Procedure: CREATE DEFINER="root"@"%" PROCEDURE
"proc_1"()
begin
select 1;
end
character_ set_ client: utf8
collation_ connection: utf8_ general_ ci
   Database Collation: utf8_ general_ ci
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: Display the statement to create a hello function.

```
gbase> show create function vc1.demo.hello\G
*************************** 1. row ***************************
                Function: hello
                sql_ mode: PIPES_ AS_ CONCAT,ANSI_ QUOTES,IGNORE_
SPACE,ONLY_ FULL_ GROUP_ BY,NO_ AUTO_ VALUE_ ON_
ZERO,STRICT_ ALL_ TABLES,NO_ ZERO_ IN_ DATE,NO_ ZERO_
DATE,NO_ AUTO_ CREATE_ USER,NO_ ENGINE_ SUBSTITUTION,PAD_
CHAR_ TO_ FULL_ LENGTH
       Create Function: CREATE DEFINER="root"@"%" FUNCTION "hello"(s
CHAR(20)) RETURNS char(50) CHARSET utf8
RETURN CONCAT('Hello, ',s,'!')
character_ set_ client: utf8
collation_ connection: utf8_ general_ ci
   Database Collation: utf8_ general_ ci
1 row in set (Elapsed: 00:00:00.00)
```

5.4.6.2 Viewing the Status of a Stored Procedure or Function

Function Description

View the status of the created or modified stored procedure or function.

Grammar format

```
SHOW {PROCEDURE | FUNCTION} STATUS;
```

Example

Example 1: Display the status of a function that has been successfully created.

```
gbase> SHOW FUNCTION STATUS\G
*************************** 1. row ***************************
                    Vc: vc1
                    Db: demo
                  Name: hello
                  Type: FUNCTION
               Definer: root@%
              Modified: 2020-07-15 19:26:08
               Created: 2020-07-15 19:26:08
      Security_ type: DEFINER
               Comment:
character_ set_ client: utf8
collation_ connection: utf8_ general_ ci
  Database Collation: utf8_ general_ ci
1 row in set (Elapsed: 00:00:00.00)
```

Example 2: Display the status of a successfully created stored procedure in the demo library of vc1.

```
gbase>    SHOW PROCEDURE STATUS where vc='vc1' and db='demo'\G
*************************** 1. row ***************************
                    Vc: vc1
                    Db: demo
                  Name: proc_ one
                  Type: PROCEDURE
               Definer: root@%
              Modified: 2020-07-15 19:25:14
               Created: 2020-07-15 19:25:14
      Security_ type: DEFINER
               Comment:
character_ set_ client: utf8
collation_ connection: utf8_ general_ ci
   Database Collation: utf8_ general_ ci
1 row in set (Elapsed: 00:00:00.00)
```

5.4.7 Statements supported by stored procedures

GBase 8a MPP Cluster not only supports the most basic structures in stored proc edures, but also supports some process control structures and statements used to i mplement specific logic in the Process Definition section. These structures and sta tements are mainly used to implement branches and loops.

5.4.7.1 DELIMITER

Grammar format

```
DELIMITER [Delimiter]
```

Table 5171 Parameter Description

Parameter Name	Description
Delimiter	Notify the client that they have completed inputting characters or string symbols for an SQL statement, typically using a semicolon ';'. In stored procedures and functions, because they contain many statements, each one requires a semicolon, it is necessary to select symbols that are unlikely to appear in the statement as separators, such as "//".

be careful

- After the stored procedure or function is defined, it is necessary to use "DELIMITER;" to restore the default terminator.

Example

Example 1: Use//as the separator.

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS dodeclare //
Query OK, 0 rows affected

gbase> CREATE PROCEDURE dodeclare (p1 INT)
            BEGIN
            DECLARE intX INT;
            SET intX = 0;
            REPEAT SET intX = intX + 1; UNTIL intX > p1 END REPEAT;
            SELECT intX;
            END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> CALL dodeclare(1000);
    +------+
    | intX |
    +------+
    | 1001 |
    +------+
    1 row in set
Query OK, 0 rows affected
```

5.4.7.2 BEGIN... END

Function Description

Stored programs (procedures and functions) may contain multiple statements, so B EGIN is used END compound statement.

Grammar format

```
[begin_label:] BEGIN
```

```
    [statement_list]
END [end_label]
```

Table 5172 Parameter Description

Parameter Name	Description
statement_ list	Represents a list of one or more statements. Separate multiple statements with a semicolon ';'.

explain

- Compound statements can be marked. end_ The label is only available at the beginning_ The label can only be used after it appears, and if both appear, they must be the same;
- To use multiple statements, the client needs to be able to send a query string containing the statement separator ';'. This can be handled by changing the command using the separator on the gbase command line on the client side. Change the delimiter ";" at the end of the query (for example, to//) to allow ";" to be used in the program body.

5.4.8 Variables of stored procedures

5.4.8.1 DECLARE

Function Description

The DECLARE statement is used to declare local variables.

explain

- DECLARE can only be used in BEGIN Between END compound statements and must precede other statements.
- The cursor must be declared before declaring the processor variable, and the condition must be declared before declaring the cursor or processor.
- The scope of action of a local variable is within its declared BEGIN Between END blocks. Variables can be used within nested blocks unless a variable with the same name is declared within the block.

Grammar format

```
DECLARE var_ name[,...] type [DEFAULT value]
```

Table 5173 Parameter Description

Parameter Name	Description

Parameter Name	Description
var_ name	Variable name
type	Variable data type with values supported by GBase 8a MPP Cluster
DEFAULT value	Set the default value of the variable. This value can be specified as an expression or a constant. If DEFAULT is missing a clause, the initial value is NULL.

Example

Example 1: DECLARE intX INT

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS dodeclare //
Query OK, 0 rows affected

gbase> CREATE PROCEDURE dodeclare (p1 INT)
       BEGIN
       DECLARE intX INT;
       SET intX = 0;
       REPEAT SET intX = intX + 1; UNTIL intX > p1 END REPEAT;
       SELECT intX;
       END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> CALL dodeclare(1000);
+------+
| intX |
+------+
| 1001 |
+------+
1 row in set
Query OK, 0 rows affected
```

Example 2: DECLARE CONTANUE HANDLER FOR SQLSTATE '02000' SET done=1

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS curdemo //
Query OK, 0 rows affected

gbase> CREATE PROCEDURE curdemo()
            BEGIN
```

```
        DECLARE done INT DEFAULT 0;
        DECLARE cnt INT DEFAULT 0;
        DECLARE s_ region CHAR(255);
        DECLARE stmp CHAR(255) DEFAULT '';
        DECLARE cur_ region CURSOR FOR SELECT DISTINCT c_
    region FROM ssbm.customer ORDER BY c_ region LIMIT 1000;
        DECLARE CONTINUE HANDLER FOR SQLSTATE '02000'
    SET done = 1;
        DROP TABLE IF EXISTS products;
        CREATE TABLE products(region CHAR(255),count INT);
        OPEN cur_ region;
        REPEAT
        FETCH cur_ region INTO s_ region;
            IF NOT done THEN
                IF   stmp=''   THEN
                    SET stmp=s_ region;
                    SET cnt=1;
                END IF;
                IF stmp!= s_ region   THEN
                    INSERT INTO products(region,count)
    VALUES(stmp,cnt);
                    SET cnt=1;
                    SET stmp=s_ region;
                END IF;
                SET cnt=cnt+1;
            END IF;
        UNTIL done END REPEAT;
        CLOSE cur_ region;
        INSERT INTO products(region,count) VALUES(stmp,cnt);
        END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> CALL curdemo;
Query OK, 1 row affected

gbase> SELECT region,count FROM products;
    +-----------------------------------------------+-------+
    | region                                        | count |
    +-----------------------------------------------+-------+
    | AFRICA                                        |     2 |
    | AMERICA                                       |     2 |
    | ASIA                                          |     2 |
    | EUROPE                                        |     2 |
```

```
| MIDDLE EAST                                                    |      2 |
+----------------------------------------------------------------+--------+
 5 rows in set
```

5.4.8.2 SET

Function Description

Used to set variable values.

explain

- The SET statement in a stored procedure is an extension of a general SET statement. The referenced variable can be declared in a stored procedure or global server variable;
- The SET statement in the stored procedure is only a partial implementation of the existing SET syntax. This allows for extended syntax SET a=x, b=y Here, different variable types can be mixed (local variables and global and session server variables). This also allows the combination of local variables and some meaningful options for system variables, in which case the options are recognized but ignored.

Grammar format

```
SET var_ name = expr [, var_name = expr]
```

Table 5174 Parameter Description

Parameter Name	Description
var_ name	Variable name
expr	Variable value

Example

Example 1: SET intX=0

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS dodeclare //
Query OK, 0 rows affected

gbase> CREATE PROCEDURE dodeclare (p1 INT)
       BEGIN
       DECLARE intX INT;
       SET intX = 0;
       REPEAT SET intX = intX + 1; UNTIL intX > p1 END REPEAT;
```

```
    SELECT intX;
    END //
Query OK, 0 rows affected

gbase> DELIMITER ;

gbase> CALL dodeclare(1000);
+------+
| intX |
+------+
| 1001 |
+------+
1 row in set
Query OK, 0 rows affected
```

5.4.8.3 SELECT... INTO...

Function Description

Store the selected column directly in a variable. Only a single row can be retrieved.

be careful

- Can SQL variable names match column names.
- If SELECT SQL statements like INTO contain a reference to a column and a local variable with the same name as the column, which interprets the reference as the name of a variable.

Grammar format

```
SELECT col_ name[,...] INTO var_ name[,...] table_ expr
```

explain

This statement stores the selected column in a variable. Only the results of a single row can be retrieved.

Example

Example 1: SELECT intX INTO @ intResult;

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS doselect_ into //
Query OK, 0 rows affected
```

```
gbase> CREATE PROCEDURE doselect_ into (p1 INT)
            BEGIN
            DECLARE intX INT;
            SET intX = 0;
            REPEAT SET intX = intX + 1; UNTIL intX > p1 END REPEAT;
            SELECT intX INTO @intResult;
            SELECT @intResult;
            END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> CALL doselect_ into (1000);
    +------------+
    | @intResult |
    +------------+
    |       1001 |
    +------------+
    1 row in set
    Query OK, 0 rows affected
```

Example 2: The column name is the same as the variable name. When this program is called, regardless of the value of the table.xname column, the variable newname will return the value 'bob'.

```
CREATE PROCEDURE sp1 (x VARCHAR(5))
  BEGIN
    DECLARE xname VARCHAR(5) DEFAULT 'bob';
    DECLARE newname VARCHAR(5);
    DECLARE xid INT;

    SELECT xname,id INTO newname,xid
      FROM table1 WHERE xname = xname;
    SELECT newname;
  END;
```

5.4.9 **Process structure supported by stored procedures**

5.4.9.1 **IF**

Function Description

The IF structure of GBase 8a MPP Cluster is a simple conditional branching structure.


explain

- The IF structure of GBase 8a MPP Cluster allows for nesting.

Grammar format

```
IF search_ condition THEN statement_ list
    [ELSEIF search_condition THEN statement_list] ...
    [ELSE statement_list]
END IF
```

Table -5175 Parameter Description

Parameter Name	Description
search_ condition	Matching criteria, if true, corresponding statement_ The list will be executed.
statement_ list	The set of SQL statements to be executed can be one statement or multiple statements.
ELSEIF search_ condition THEN statement_ list	Matching process branches, if the previous search_ The condition does not match, the statement list in ELSEDIF will continue.
ELSE statement_ list	If all the previous searches_ The conditions do not match, and the statement list in the ELSE clause will be executed.

Example

Example 1: IF THEN... ELSE... END IF。

```
gbase> DELIMITER //
gbase> DROP FUNCTION IF EXISTS fn_ count //
Query OK, 0 rows affected

gbase> CREATE FUNCTION fn_ count (param    VARCHAR(10))
RETURNS INT
        BEGIN
        SELECT COUNT(*)/3 INTO @count FROM ssbm.customer
WHERE c_ nation= 'JORDAN';
        IF @count<=3 THEN
        RETURN @count;
        ELSE
        RETURN @count/3;
        END IF;
          END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> SET @result = fn_ count ('JORDAN');
```

```
Query OK, 0 rows affected

gbase> SELECT @result;
+---------+
| @result |
+---------+
|       131 |
+---------+
1 row in set
```

5.4.9.2 **ITERATE**

Function Description

The ITERATE statement is used to achieve repeated execution back to the specified position. It can only appear in LOOP, REPEAT, and WHILE structures, and the label of the position to return to must be defined for the statement. The label must then be specified where the statement is used. The ITERATE statement is usually placed in an IF structure to achieve repeated execution based on conditions.

Grammar format

```
ITERATE<label>
```

Example

Example 1: ITERATE

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS doiterate //
Query OK, 0 rows affected

gbase> CREATE PROCEDURE doiterate(p1 INT)
        BEGIN
        label1: LOOP
        SET p1 = p1 + 1;
        IF p1 < 10 THEN ITERATE label1; END IF;
        LEAVE label1;
        END LOOP label1;
        SET @x = p1;
        END //
Query OK, 0 rows affected

gbase> DELIMITER ;
```

```
gbase> CALL doiterate(1);
Query OK, 0 rows affected

gbase> SELECT @x;
+------+
| @x   |
+------+
|   10 |
+------+
1 row in set
```

5.4.9.3 CASE

Function Description

GBase 8a MPP Cluster uses the CASE structure to handle the situation of multiple branches.

be careful

- CASE calculations also rely on context. If it is a string context, the returned result is treated as a string. If it is a numerical context, the returned result is a numerical value.

Grammar format

Grammar Format 1

```
CASE case_ value
      WHEN when_ value THEN statement_ list
     [WHEN when_value THEN statement_list] ...
      [ELSE statement_list]
END CASE;
```

Table 5176 Parameter Description

Parameter Name	Description
Case_ value	Value to be matched
When_ value	Matching value, if matched, the corresponding statement_ The list statement will be executed.
statement_ list	The set of SQL statements to be executed can be one statement or multiple statements.
ELSE statement_ list	If all the previous when_ No values match, the statement list in

Parameter Name	Description
	the ELSE clause will be executed.

Grammar Format 2

```
CASE
    WHEN search_ condition THEN statement_ list
    [WHEN search_condition THEN statement_list] ...
    [ELSE statement_list]
END CASE;
```

Table 5177 Parameter Description

Parameter Name	Description
search_ condition	Matching criteria, if true, corresponding statement_ The list will be executed.
statement_ list	The set of SQL statements to be executed can be one statement or multiple statements.
ELSE statement_ list	If all the previous searches_ The conditions do not match, and the statement list in the ELSE clause will be executed.

Example

Example 1: No Case After Case_ value。

```
gbase> delimiter //

gbase> CREATE PROCEDURE casedemo(in para int,out x varchar(100))

    begin

      case

        when para=1 then set x='true';

        when para=0 then set x='false';

        else set x='error';

      end case;

    end //

Query OK, 0 rows affected (Elapsed: 00:00:00.03)



gbase> delimiter ;
```

```
gbase> call casedemo(1,@result);

Query OK, 0 rows affected (Elapsed: 00:00:00.03)



gbase> select @result;

+---------+

| @result |

+---------+

| true    |

+---------+

1 row in set (Elapsed: 00:00:00.00)
```

Example 2: Case after Case_ value。

```
gbase> DELIMITER //
gbase> CREATE PROCEDURE casedemo(in para int,out x varchar(100))

        begin

          case para

             when 1 then set x='true';

             when 0 then set x='false';

             else set x='error';

           end case;

      end //

Query OK, 0 rows affected (Elapsed: 00:00:00.08)



gbase> delimiter ;

gbase> call casedemo(1,@result);

Query OK, 0 rows affected (Elapsed: 00:00:00.02)
```

```
gbase> select @result;
+---------+
| @result |
+---------+
| true    |
+---------+
1 row in set (Elapsed: 00:00:00.00)

gbase> call casedemo(2,@result);
Query OK, 0 rows affected (Elapsed: 00:00:00.03)

gbase> select @result;
+---------+
| @result |
+---------+
| error   |
+---------+
1 row in set (Elapsed: 00:00:00.00)
```

5.4.9.4 LOOP

Function Description

The LOOP structure is a simple loop structure in GBase 8a MPP Cluster, used to repeatedly execute a statement or a set of statements. This loop structure is formally a dead loop structure, so it usually includes a conditional judgment statement and a LEAVE statement in the execution body to exit the loop.

Grammar format

```
[begin_label:] LOOP
        statement_ list
END LOOP [end_label]
```

Table -5178 Parameter Description

Parameter Name	Description
statement_ list	The set of SQL statements to be executed can be one statement or multiple statements.

Example

Example 1: LOOP END LOOP

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS doiterate //
Query OK, 0 rows affected

gbase> CREATE PROCEDURE doiterate(p1 INT)
            BEGIN
            label1: LOOP
            SET p1 = p1 + 1;
            IF p1 < 10 THEN ITERATE label1; END IF;
            LEAVE label1;
            END LOOP label1;
            SET @x = p1;
            END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> CALL doiterate(1);
Query OK, 0 rows affected

gbase> SELECT @x;
+------+
| @x   |
+------+
|   10 |
+------+
1 row in set
```

5.4.9.5 REPEAT

Function Description

The REPEAT structure is a common loop structure in GBase 8a MPP Cluster, which repeatedly executes the execution body until the exit condition is met.


explain

- The execution body of the REPEAT structure will be executed at least once. If this is not allowed, the WHILE structure can be used instead.

Grammar format

```
[begin_label:] REPEAT
    statement_ list
UNTIL search_ condition
END REPEAT [end_label]
```

Table 5179 Parameter Description

Parameter Name	Description
statement_ list	The set of SQL statements to be executed can be one statement or multiple statements.
search_ condition	The end condition of the REPEAT statement, if true, the REPEAT statement will end

Example

Example 1: REPEAT UNTIL... END REPEAT

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS dorepeat //
Query OK, 0 rows affected

gbase> CREATE PROCEDURE dorepeat(p1 INT)
    BEGIN
    SET @x = 0;
    REPEAT SET @x = @x + 1; UNTIL @x > p1 END REPEAT;
    END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> CALL dorepeat(1000);
Query OK, 0 rows affected
```

```
gbase> SELECT @x;
+------+
| @x   |
+------+
| 1001 |
+------+
1 row in set
```

5.4.9.6 **WHILE**

Function Description

WHILE is another common loop structure in GBase 8a MPP Cluster, which repeatedly executes the execution body when the execution conditions are met.

explain

- The WHILE structure is logically consistent with REPEAT, except that the executor in the REPEAT structure will execute at least once, while the executor in the WHILE structure may not execute at all.

Grammar format

```
[begin_label:] WHILE search_ condition DO
    statement_ list
END WHILE [end_label]
```

Table 5180 Parameter Description

Parameter Name	Description
statement_ list	The set of SQL statements to be executed can be one statement or multiple statements.
search_ condition	The WHILE statement end condition, if true, the WHILE statement will end

Example

Example 1: WHILE END WHILE

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS doWhile //
Query OK, 0 rows affected

gbase> CREATE PROCEDURE doWhile(p1 INT)
    BEGIN
    SET @x = 0;
```

```
        WHILE    @x < p1 DO    SET @x = @x + 1; END WHILE;
         END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> CALL dowhile(1000);
Query OK, 0 rows affected

gbase> SELECT @x;
+------+
| @x     |
+------+
| 1000 |
+------+
1 row in set
```

5.4.9.7 LEAVE

Function Description

The LEAVE statement in GBase 8a MPP Cluster is used to exit the loop structure, so it can only appear in LOOP, REPEAT, and WHILE structures. The LEAVE statement is usually placed in the IF structure to achieve conditional exit of the loop structure. Similarly, when using the LEAVE statement, a label must be defined for the loop structure containing the statement, and then specified where the statement is used.

Grammar format

```
LEAVE label
```

Example

Example 1: LEAVE

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS doiterate //
Query OK, 0 rows affected

gbase> CREATE PROCEDURE doiterate(p1 INT)
           BEGIN
           label1: LOOP
           SET p1 = p1 + 1;
           IF p1 < 10 THEN ITERATE label1; END IF;
           LEAVE label1;
```

```
        END LOOP label1;
        SET @x = p1;
        END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> CALL doiterate(1);
Query OK, 0 rows affected

gbase> SELECT @x;
+------+
| @x   |
+------+
|   10 |
+------+
1 row in set
```

5.4.10 cursor

5.4.10.1 Static cursor (CURSOR)

summary

The result set returned by a SELECT statement typically includes a series of record rows, but there are often cases where the entire result set may not be effectively processed as a unit. At this point, a mechanism is needed to process one row of records at a time, and cursors in the database provide this mechanism.

The definition and use of cursors in the GBase 8a MPP Cluster database have the following limitations:

- Static cursor, which means that the result set of the SELECT STATION statement must be specified for binding during DECLARE; In subsequent operations, only read-only and forward only operations can be performed on the result set;
- The cursor must be declared before declaring the processor, and variables and conditions must be declared before declaring the cursor or processor.

5.4.10.1.1 Definition of Static Cursor

Function Description

In GBase 8a MPP Cluster, DECLARE is used to define cursors, and the annotated body must be a SELECT statement.

be careful

- Multiple cursors can be defined in a program, but cursors in each block can only have unique names;
- A SELECT statement cannot contain an INTO clause.

Grammar format

```
DECLARE cursor_ name CURSOR FOR <select_ statement>
```

Table 5181 Parameter Description

Parameter Name	Description
cursor_ name	The name of the cursor to be created, which only allows a-z, A-Z, 0-9, underscores, and cannot contain only numbers;
select_ statement	The content of the cursor can be any valid SELECT statement

Example

```
DECLARE cur CURSOR FOR SELECT DISTINCT lo_ orderkey FROM
ssbm.lineorder ORDER BY lo_ orderkey LIMIT 10;
```

5.4.10.1.2 Using static cursors

Function Description

The purpose of using a cursor is to obtain the values of the fields in the result set returned by the SELECT statement in the cursor definition. In GBase 8a MPP Cluster, this value taking process is also achieved through the FETCH statement.

Grammar format

```
FETCH cursor_ name INTO var_ name [, var_name] ...
```

Table 5182 Parameter Description

Parameter Name	Description
cursor_ name	The name of the previously defined cursor needs to obtain t he return value from it.
var_ name	The name of a local variable and the value obtained from the cursor should be stored in these local variables. The FETCH statement requires the number of local variables to be the same as the number of fields in the selection list in the SELECT statement of the cursor definition statement, and the data type should also correspond to the same or can be automatically converted. These local variables will be processed in subsequent statements.

Example

The following code is included in the cursor code block.

```
DECLARE s_ region CHAR(16);
DECLARE region INT;
DECLARE cur CURSOR FOR SELECT DISTINCT c_ region,1 FROM
ssbm.customer    ORDER BY c_ region LIMIT 1000;
OPEN cur;
FETCH cur INTO s_ region, region; //
```

5.4.10.1.3 Close Static Cursor

Function Description

The cursor needs to be closed after use, otherwise the server resources occupied by the cursor will not be released; If there is no explicit closure, the cursor is closed at the end of the compound statement declaring it.

Grammar format

```
CLOSE cursor_ name
```

Table 5183 Parameter Description

Parameter Name	Description
cursor_ name	The name of the cursor to close.

Example

The following code is included in the cursor code block.

```
DECLARE s_ region CHAR(16);
DECLARE region    INT;
DECLARE cur CURSOR FOR SELECT DISTINCT c_ region,1 FROM
ssbm.customer ORDER BY c_ region LIMIT 1000;
OPEN cur;
FETCH cur INTO s_ region, region;
CLOSE cur; //
```

5.4.10.1.4 Precautions for using static cursors

- The cursor in GBase 8a MPP Cluster is a read-only, forward only cursor. Th e data contained in the cursor cannot be changed during use, and the data in the cursor can only be read in order from beginning to end;
- The cursor in GBase 8a MPP Cluster needs to be used in conjunction with t he processor. The cursor needs to be declared before the processor's declarati on statement, and any variables used within the cursor need to be defined be fore the cursor's declaration statement;
- When using cursors to process data, it is common to use LOOP, REPEAT, or WHILE structures, and use FETCH statements in the execution body of t hese structures to traverse the data in the cursor;
- In GBase 8a MPP Cluster, multiple cursors can be declared within the same stored procedure, but with the following limitations:
  1. Multiple cursors cannot cross each other, it is best to use one before usin g another;
  2. If the LOOP, REPEAT, or WHILE structures are used to traverse the cur sor to obtain data and process it, and if the stored procedure is called in the structure of these loop structures, the called stored procedure should no longer contain the cursor and the LOOP, REPEAT, or WHILE structu res used to traverse the cursor, or unexpected results may occur.

5.4.10.1.5 Example of a static cursor

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS docursor //
Query OK, 0 rows affected

gbase> CREATE PROCEDURE docursor()
```

```
        BEGIN
        DECLARE s_ region VARCHAR(40);
        DECLARE DONE INT DEFAULT(0);
        DECLARE cur CURSOR FOR SELECT DISTINCT c_ region FROM
ssbm.customer ORDER BY c_ region LIMIT 6;
        DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET
done = 1;
        OPEN cur;
        REPEAT
        FETCH cur    INTO s_ region;
        IF NOT done THEN
        SELECT s_ region;
        END IF;
        UNTIL DONE END REPEAT;
        CLOSE cur;
        END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> CALL docursor();
+----------+
| s_ region |
+----------+
| AFRICA     |
+----------+
1 row in set

+----------+
| s_ region |
+----------+
| AMERICA    |
+----------+
1 row in set

+----------+
| s_ region |
+----------+
| ASIA         |
+----------+
1 row in set

+----------+
| s_ region |
+----------+
```

```
| EUROPE     |
+----------+
1 row in set

+-------------+
| s_ region       |
+-------------+
| MIDDLE EAST |
+-------------+
1 row in set

Query OK, 0 rows affected
```

5.4.10.2 Dynamic Cursor (REF CURSOR)

summary

GBase 8a MPP Cluster supports the definition and use of dynamic cursors; As an enhancement of static cursors, declaring REF CURSOR as a dynamic cursor during DECLARE allows for multiple binding of result sets of different SELECT STATION statements during OPEN.

5.4.10.2.1 Definition of Dynamic Cursor

grammar

```
DECLARE cursor_ name REF CURSOR
```

Table 5184 Parameter Description

Parameter Name	Description
cursor_ name	The name of the cursor to be created, which only allows a-z, A-Z, 0-9, underscores, and cannot contain only numbers;

be careful

- When defining dynamic cursors using DECLARE, it is not allowed to specify any SELECT statement.

Example

```
DECLARE cur REF CURSOR;
```

5.4.10.2.2 Open Dynamic Cursor

Function Description

Just like using static cursors, you also need to use the OPEN statement to open the cursor before using dynamic cursors.

grammar

```
OPEN <cursor_ name> FOR <select_ statement>
```

Table 5185 Parameter Description

Parameter Name	Description
cursor_ name	The name of the cursor to open
select_ statement	The content of a cursor can be any valid SELECT statement or a dynamic SQL statement.

Example

The following code is included in the cursor code block.

```
DECLARE cur REF CURSOR;
OPEN cur FOR SELECT DISTINCT c_ region,1 FROM ssbm.customer ORDER BY c_ region LIMIT 1000;
```

5.4.10.2.3 Using dynamic cursors

Function Description

By using the FETCH statement, the values of the fields in the result set returned by the SELECT statement in the dynamic cursor OPEN statement can be obtained.

grammar

```
FETCH cursor_ name INTO var_ name [, var_name] ...
```

Table 5186 Parameter Description

Parameter Name	Description
cursor_ name	The name of the cursor opened through OPEN.
var_ name	The name of a local variable and the value obtained from the

Parameter Name	Description
	cursor should be stored in these local variables. The FETCH statement requires the number of local variables to be the same as the number of fields in the select list in the dynamic cursor OPEN statement, and the data type should also correspond to the same or can be automatically converted.

Example

The following code is included in the cursor code block.

```
DECLARE s_ region CHAR(16);
DECLARE region INT;
DECLARE cur REF CURSOR;
OPEN cur FOR SELECT DISTINCT c_ region,1 FROM ssbm.customer
ORDER BY c_ region LIMIT 1000;
FETCH cur INTO s_ region, region; //
```

5.4.10.2.4 Turn off dynamic cursors

Function Description

The cursor needs to be closed after use, otherwise the server resources occupied by the cursor will not be released; If there is no explicit closure, the cursor is closed at the end of the compound statement declaring it.

grammar

```
CLOSE cursor_ name
```

Table 5187 Parameter Description

Parameter Name	Description
cursor_ name	The name of the cursor to close.

Example

Example 1: The following code is included in the cursor code block.

```
DECLARE s_ region CHAR(16);
DECLARE region INT;
DECLARE cur REF CURSOR;
OPEN cur FOR SELECT DISTINCT c_ region,1 FROM ssbm.customer
ORDER BY c_ region LIMIT 1000;
FETCH cur INTO s_ region, region; //
CLOSE cur; //
```

5.4.10.2.5 Dynamic SQL syntax used in dynamic cursors

Function Description

Through dynamic SQL, the SELECT statement in the dynamic cursor OPEN statement can be represented by a text string or a user variable whose content is a text string.

grammar

```
OPEN cursor_ name FOR select_ statement
```

Table -5188 Parameter Description

Parameter Name	Description
cursor_ name	Dynamic cursor name
select_ statement	Open a dynamic SQL statement with a dynamic cursor.

Example

Example 1: SELECT * From hunter. t1

```
DECLARE cur REF CURSOR;
SET v = 'SELECT * FROM hunter.t1';
SET @sql_ str = v;
OPEN cur FOR @sql_ str;
FETCH cur INTO i, j;
```

5.4.10.2.6 Precautions for using dynamic cursors

When using dynamic cursors, the following precautions should be taken:

- When a dynamic cursor is in DECLARE, it is not allowed to specify any S ELECT statement.
- Dynamic cursors allow nesting. If there is an OPEN operation within the nes ting, it must be within the same nesting

  The Close operation and the OPEN operation appear in pairs to avoid errors caused by repeated OPEN operations.
- Dynamic cursors can only be used in stored procedures.

5.4.10.2.7 Dynamic Cursor Example

Example 1: The query statement is a static SQL statement.

```
gbase> DELIMITER //
gbase> DROP PROCEDURE IF EXISTS docursor //
```

```
Query OK, 0 rows affected

gbase> CREATE PROCEDURE docursor()
       BEGIN
       DECLARE s_ region VARCHAR(40);
       DECLARE DONE INT DEFAULT(0);
       DECLARE cur REF CURSOR;
       DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET done = 1;
       OPEN cur FOR SELECT DISTINCT c_ region FROM ssbm.customer ORDER BY c_
region LIMIT 6;
       REPEAT
       FETCH cur    INTO s_ region;
       IF NOT done THEN
       SELECT s_ region;
       END IF;
       UNTIL DONE END REPEAT;
       CLOSE cur;
       END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> CALL docursor();
+----------+
| s_ region |
+----------+
| AFRICA     |
+----------+
1 row in set

+----------+
| s_ region |
+----------+
| AMERICA    |
+----------+
1 row in set

+----------+
| s_ region |
+----------+
| ASIA       |
+----------+
1 row in set

+----------+
```

```
| s_ region |
+----------+
| EUROPE     |
+----------+
1 row in set

+-------------+
| s_ region       |
+-------------+
| MIDDLE EAST |
+-------------+
1 row in set

Query OK, 0 rows affected
```

Example 2: The SELECT statement in the OPEN statement contains a user variable consisting of a text string or a content that is a text string.

Tables and data used in the example:

```
DROP TABLE t1;
CREATE TABLE t1 (i INT, j INT);
INSERT INTO t1 VALUES(1, 1);
INSERT INTO t1 VALUES (1, 1);
INSERT INTO t1 VALUES (2, 2);
INSERT INTO t1 VALUES (3, 3);
INSERT INTO t1 VALUES (4, 4);
SELECT * FROM t1;
```

Create a stored procedure:

```
gbase> DELIMITER //
gbase> CREATE PROCEDURE hunter.test_ 1()
        BEGIN
          DECLARE v VARCHAR(200);
          DECLARE i INT DEFAULT (0);
          DECLARE j INT DEFAULT (0);
          DECLARE cur REF CURSOR;
          SET v = 'SELECT * FROM hunter.t1';
          SET @sql_ str = v;
          OPEN cur FOR @sql_ str;
          FETCH cur INTO i, j;
          SELECT i, j;
          CLOSE cur;
        END //
Query OK, 0 rows affected
```

```
Execution results:
gbase> DELIMITER ;
gbase> CALL hunter.test_ 1();
+------+------+
| i       | j       |
+------+------+
|       1 |       1 |
+------+------+
1 row in set

Query OK, 0 rows affected
```

Example 3: A preprocessing statement in a dynamic cursor contains a dynamic SQL statement.

```
gbase> DELIMITER //
gbase> CREATE PROCEDURE hunter.test_ 1()
        BEGIN
          DECLARE v VARCHAR(200);
          SET v = 'SELECT * FROM hunter.t1 WHERE i = ? AND j = ?';
          SET @sql_ str = v;
          SET @a = 1;
          SET @b = 2;
          PREPARE stmt FROM @sql_ str;
          EXECUTE stmt USING @a, @b;
        END //
Query OK, 0 rows affected

gbase> DELIMITER ;
gbase> CALL hunter.test_ 1();
Empty set

Query OK, 0 rows affected
```

5.4.11 Stored Procedure Exception Handling

5.4.11.1 Conditions and processing procedures

5.4.11.1.1 DECLSTR conditional declaration

Function Description

Specific processing used to define specific conditions.

Grammar format

```
DECLARE condition_ name CONDITION FOR condition_ value

condition_ value:
      SQLSTATE [VALUE] sqlstate_ value
   | gbase_ error_ code
```

Table 5189 Parameter Description

Parameter Name	Description
condtion_ name	Conditional naming, regular variable naming
condition_ value	SQLState value or gbase_ error_ code

5.4.11.1.2 DECLSTR conditional processing

Function Description

The statement specifies the handler, each capable of handling one or more conditions. If one or more conditions are generated, the specified statement will be executed.

Grammar format

```
DECLARE handler_ type HANDLER FOR condition_ value[,...] statement

handler_ type:
      CONTINUE
   | EXIT

condition_ value:
      SQLSTATE [VALUE] sqlstate_ value
   | condition_ name
   | SQLWARNING
   | NOT FOUND
```

```
  | SQLEXCEPTION
  | gbase_ error_ code
```

Table -5190 Parameter Description

Parameter Name	Description
handler_ type	Actions processed
CONTINUE	After the processor statement execution ends, the current program continues to execute.
EXIT	Current BEGIN The execution of the END compound statement was terminated.
statement	One or more statements
condition_ value	Exception handling capture conditions or situations
condtion_ name	Condition name, using DECLARE To define using the CONDITION statement
SQLWARNING	Short hand for all SQLSTATE codes starting with 01
NOT FOUND	Short hand for all SQLSTATE codes starting with 02
SQLEXCEPTION	A shorthand for all SQLSTATE code that has not been captured by SQLWARNING or NOTFOUND.

Example

Example 1:

```
DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET
has_ error = 1;
```

5.4.11.2 GET DIAGNOSTICS

Function Description

The statement is used to obtain the content of the error buffer, including error information, DML operation impact results, etc

Grammar format

```
GET [CURRENT | STACKED] DIAGNOSTICS
{
      statement_ information_ item [, statement_information_item] ...
  | CONDITION condition_ number condition_ information_ item
      [, condition_information_item] ...
}

statement_ information_ item:
      target = statement_ information_ item_ name
```

```
condition_ information_ item:
        target = condition_ information_ item_ name

statement_ information_ item_ name:
        NUMBER
    | ROW_ COUNT

condition_ information_ item_ name: {
        CLASS_ ORIGIN
    | SUBCLASS_ ORIGIN
    | RETURNED_ SQLSTATE
    | MESSAGE_ TEXT
    | GBASE_ ERRNO
    | CONSTRAINT_ CATALOG
    | CONSTRAINT_ SCHEMA
    | CONSTRAINT_ NAME
    | CATALOG_ NAME
    | SCHEMA_ NAME
    | TABLE_ NAME
    | COLUMN_ NAME
    | CURSOR_ NAME
}
```

Table 5191 Parameter Description

Parameter Name	Description
statement_ information_ item	Capture feedback on the execution status of the statement, including NUMBER and ROW_ COUNT
condition_ information_ item	Capture abnormal situation information.

Table 5192 statement_ information_ item_ Name parameter description

Parameter Name	Description
number	The number contains the number of WARNINGS and errors.
row_ count	Only record the number of affected rows of the last DML operation before the GET DIAGNOSTICS command, and cannot accumulate. If you want to obtain the number of affected rows of multiple DML statements, you need to execute the GET DIAGNOSTICS command after each DML statement.

Table 5193 Condition_ information_ item_ Name parameter description

Parameter Name	Description
no	The serial number representing an error or WARNINGS.
number	The number contains the number of WARNINGS and errors.
row_ count	Only record the number of affected rows of the last DML

Parameter Name	Description
	operation before the GET DIAGNOSTICS command, and cannot accumulate. If you want to obtain the number of affected rows of multiple DML statements, you need to execute the GET DIAGNOSTICS command after each DML statement.
gbase_ errno	Record the error number.
returned_ sqlstate	Record the error status.
message_ text	Record error information.

Example

Example 1: Obtain the number of errors.

```
DROP TABLE IF EXISTS t;
DROP TABLE IF EXISTS tt;
DROP PROCEDURE IF EXISTS p1;
DELIMITER //
CREATE PROCEDURE p1()
BEGIN
DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
 BEGIN
 GET DIAGNOSTICS @num1 = NUMBER;
 SELECT @num1;
 END;
 DELETE FROM tt ;
 DROP TABLE IF EXISTS t;
 GET DIAGNOSTICS @num2 = NUMBER;
 SELECT @num2;

END //
DELIMITER ;
CALL p1;
```

The execution results are as follows:

```
gbase> CALL p1;
+-------+
| @num1 |
+-------+
| 1 |
+-------+
1 row in set
+-------+
| @num2 |
+-------+
| 1 |
+-------+
```

```
1 row in set
Query OK, 0 rows affected, 8 warnings
gbase> SHOW WARNINGS; +-------+------+----------------------------------------+
| Level | Code | Message |
+-------+------+----------------------------------------+
| Note | 1051 | 192.168.103.77:5050 - Unknown table 't' |
| Note | 1051 | 192.168.103.75:5050 - Unknown table 't' |
| Note | 1051 | 192.168.103.74:5050 - Unknown table 't' |
| Note | 1051 | 192.168.103.76:5050 - Unknown table 't' |
| Note | 1051 | 192.168.103.77:5258 - Unknown table 't' |
| Note | 1051 | 192.168.103.75:5258 - Unknown table 't' |
| Note | 1051 | 192.168.103.76:5258 - Unknown table 't' |
| Note | 1051 | 192.168.103.74:5258 - Unknown table 't' |
+-------+------+----------------------------------------+
8 rows in set
```

Example 2: Obtain the number of rows affected by the DML operation. After the completion of a non DML operation, if the GET DIAGNOSTICS syntax is used to obtain the number of rows affected by the DML operation, the result obtained is -1.

```
DROP PROCEDURE IF EXISTS p1;
DELIMITER |
CREATE PROCEDURE p1()
BEGIN
 DECLARE row_ count INT;

 GET DIAGNOSTICS row_ count = ROW_ COUNT;
 SELECT row_ count;

END|
DELIMITER ;
CALL p1();
The execution results are as follows:
gbase> CALL p1();
+-----------+
| row_ count |
+-----------+
| -1 |
+-----------+
1 row in set
```

Example 3: Obtain the number of rows affected by the DML operation. After the DML operation is completed, if the GET DIAGNOSTICS syntax is used to obtain the number of rows affected by the DML operation, the result obtained is 1.

```
DROP TABLE IF EXISTS tt;
CREATE TABLE t(a int);
DROP PROCEDURE IF EXISTS p1;
DELIMITER |
CREATE PROCEDURE p1()
BEGIN
 DECLARE row_ count INT;

 INSERT INTO t VALUES(1);

GET DIAGNOSTICS row_ count = ROW_ COUNT;
 SELECT row_ count;
END|
DELIMITER ;
CALL p1();
```

The execution results are as follows:

```
gbase> CALL p1();
+-----------+
| row_ count |
+-----------+
| 1 |
+-----------+
1 row in set
Query OK, 0 rows affected
```

Example 4: Obtaining error codes, error status, and error information.

```
DROP TABLE IF EXISTS t1;
CREATE TABLE t1(i int,vc varchar(20))distributed by('i');
DROP PROCEDURE IF EXISTS p1;
DELIMITER |
CREATE PROCEDURE p1()
BEGIN
DECLARE errno varchar(50);
DECLARE sstate varchar(50);
DECLARE message varchar(50);
DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
BEGIN
GET DIAGNOSTICS CONDITION 1
errno=gbase_ errno,sstate=returned_ sqlstate,message=message_ text;
END;
DELETE FROM t1 WHERE a>0;
SELECT errno,sstate,message;
END |
DELIMITER ;
```

```
CALL p1();
The execution results are as follows:
gbase> CALL p1(); +-------+--------+--------------------------------------+
| errno | sstate | message |
+-------+--------+--------------------------------------+
| 1054 | 42S22 | Unknown column 'a' in 'where clause' |
+-------+--------+--------------------------------------+
1 row in set
Query OK, 0 rows affected
```

5.4.12 Usage restrictions

Function usage restrictions:

1. DML, DDL, and creating temporary tables in functions are not supported;
2. SQL query statements involved in functions are not supported;
3. The use of the following methods to assign values to variables in functions is not supported, but is not prohibited;

   DECLARE res int DEFAULT 0;

   SET res=(SELECT COUNT(*) FROM t);

   Or: SELECT COUNT (*) INTO @ res From t;
4. Preprocessing is not supported.

SQL Scenario Usage Restrictions:

There are subqueries in SQL and there is no from clause in the subquery, no tables are queried, only the projection part. For example:

select * from t1 where b > (select now()) ; -- The sub query (select now()) does not have a from clause and does not query any tables

1. The definition in the stored procedure includes SQL for this scenario. When executing the stored procedure, an expected error was reported and is not supported;
2. SQL preprocessing execution for this scenario is not supported;
3. SQL execution for this scenario is not supported in the event, and an error was expected during event scheduling and execution, which is not supported.

Example:

```
Example of stored procedure:
DELIMITER //
CREATE  PROCEDURE testpro ()
BEGIN
select * from t1 where b > (select now()) ;  -- The sub query (select now()) did
```

```
not query any tables
END //
DELIMITER ;
call testpro(); -- Expected error
ERROR 1149 (42000): (GBA-02SC-1001) The query includes syntax that is not supported by the gcluster.
{STATEMENT: call testpro()
INSTRUCTION: select * from t where d > (select now()) [1]}
```

Prepare example:

```
gbase> delimiter //
gbase> create procedure p1()
        begin
        set @sql_ str='select * from t where d > (select now())';
        prepare stmt from @sql_ str;
        execute stmt;
        end //
Query OK, 0 rows affected (Elapsed: 00:00:00.05)
gbase> delimiter ;
gbase> call p1();
ERROR 1149 (42000): (GBA-02SC-1001) The query includes syntax that is not supported by the gcluster.
{STATEMENT: call p1()
INSTRUCTION: execute stmt [3]}
```

Event example:

```
create event myEvent2
on schedule
every 10 second
starts current_ timestamp + interval 20 second
ends current_ timestamp + interval 1 minute
on completion not preserve
do insert into t2 select * from t1 where b > (select now());  -- The sub query (select now()) did not query any tables, and an expected error was reported during event scheduling execution: (GBA-02SC-1001) The query includes syntax that is not supported by the gcluster
```

5.5 Cluster expansion

5.5.1 UDF&UDAF

5.5.1.1 summary

GBase 8a MPP Cluster supports a universal extension mechanism for UDF and UDAF, through which database users can define and develop efficient SQL functions (implemented in C/C++language).

be careful

UDF exists in the form of dynamic libraries, and its stability can affect the stability of database services.

5.5.1.2 Environmental requirements

- The operating system must support dynamic loading;
- Function implementation must use C or C++language.

5.5.1.3 UDF function interface

For each function that you want to use in an SQL statement (assuming the function name is func), you should define the corresponding C (or C++, with extern "C" added to the C++function declaration) function, which meets the following rules:

Func() (required)

Main function. This is where the function result is calculated, called once per line. The correspondence between SQL types and C/C++function return types is as follows:

Table -5194 Correspondence between SQL Types and C/C++Function Return Types

SQL Type	C/C++type

SQL Type	C/C++type
STRING	char *
INTEGER	long long
REAL	double

func_ Init() (required)

The initialization function of func() is only called once at the beginning and can be used to:

- Check the number of parameters passed to func();
- Check if the parameter type is correct or cast the parameter to the required class when the main function is called

  Type;
- Allocate the memory required for the main function;
- Specify the maximum length of the returned result;
- Specify the maximum number of decimal places to return functions of type REAL;
- Specify whether the result is allowed to be NULL.

func_ Deinit() (optional)

The end function of func() is called only once after all lines are ended. It can be used to release the memory allocated by the initialization function.

explain

- When a SQL statement calls func(), GBase calls the initialization function func_ Init() to perform the required initialization work, such as parameter checking or memory allocation.
- If func_ Init() returns an error, and the SQL statement returns an error message without calling both the main and closing functions. Otherwise, call the main function func() once per line.
- After all lines are processed, call the end function func_ Deinit(), perform necessary cleaning work.

be careful

All functions must be thread safe (not only the main function, but also the initialization and termination functions). It is not allowed to change globally shared or static variables in a function. If memory is needed, it should be stored in Func_ Allocate it in init() and in func_ Release it in deinit().

5.5.1.4 Function parameter form and structure

5.5.1.4.1 Function parameter form

- The difference in the return type and parameters of the main function depends on the SQL declared in the CREATE JUNCTION statement

The function func() returns the type.

•The function that returns STRING in SQL has the following form:

```
char *func(UDF_INIT *initid, UDF_ARGS *args,
                    char *result, unsigned long *length,
                            char *is_ null, char *error);
```

•Return the INTEGER function in SQL, as follows:

```
long long func(UDF_INIT *initid, UDF_ARGS *args,
                                    char *is_ null, char *error);
```

•Return the REAL function in SQL, as follows:

```
double func(UDF_INIT *initid, UDF_ARGS *args,
                                    char *is_ null, char *error);
```

- The declaration forms of the initialization and termination functions are as follows:

```
my_ bool func_ init(UDF_INIT *initid, UDF_ARGS *args, char *message);

void func_ deinit(UDF_INIT *initid);
```

5.5.1.4.2 Parameter structure

5.5.1.4.2.1Initid parameter

This parameter is passed to all three functions, pointing to a UDF_ The INIT structure is used to transfer information between functions. UDF_ The members of the INIT structure are listed below. The initialization function should initialize any member it wants to change. Use the default value for a member without changing it.

my_ bool maybe_ null

If func() can return NULL, func_ Init() should be set to 'may'_ Null is 1. If any parameter is declared as possible_ Null, the default value is 1.

unsigned int decimals

Number of decimal places. The default value is the maximum number of decimal places in the parameter passed to the main function. For example, if the function passes 1.11, 1.111, and 1.1, the default value will be 3 because 1.111 has 3 decimal places.

unsigned int max_ length

The maximum length of the returned result. The default values vary depending on the result type of the function. For string functions, the default is the length of the longest argument. For integer functions, the default is 21 bits. For real number functions, the default is 13 plus the number of decimal places indicated by initid ->decimals. For numeric functions, the length includes any sign or decimal character.

char *ptr

A pointer that a function can use on its own. For example, functions can use initid ->ptr to pass allocated memory between functions. At Func_ In init(), allocate memory and assign it to this pointer:

```
initid->ptr = allocated_ memory;
```

 **explain**

In func() and func_ In deinit(), use initid ->ptr and release memory.

5.5.1.4.2.2Args parameter

This parameter points to a UDF_ The structure of ARGS members is as follows:

unsigned int arg_ count

Number of parameters. If the function has a fixed number of parameters, check this value in the initialization function. For example:

```
if (args->arg_count != 1)
{
        strcpy(message," func() requires one arguments");
        return 1;
}
```

enum Item_ result *arg_ type

- The type of each parameter. The possible type value is STRING_ RESULT、INT_ RESULT and

  REAL_ RESULT;

- Ensure that the parameter is of the required type, and if not, return an error;

- Check arg in the initialization function_ Type array. For example:

```
if (args->arg_type[0] != STRING_RESULT
                && args->arg_ type[1] != INT_ RESULT)
{
        strcpy(message," func() requires a string and an integer");
        return 1;
}
```

explain

You can also use the initialization function to set arg_ Set the type member to the desired type. This way, GBase calls strong

Convert parameters to the desired type. For example, to specify that the first two parameters are strings and integers, you can use func_ init()

To do this:

```
args->arg_ type[0] = STRING_ RESULT;
```

```
args->arg_ type[1] = INT_ RESULT;
```

char **args

Args passes the parameters of the function to the initialization function and the main function. The way the function references the i-th parameter is as follows:

- A STRING_ A parameter of type RESULT is given by a string pointer and a length, allowing for

  Processing data of any length;

- The string content can be obtained from args ->args [i] and the string length is args ->lengths [i]. No need to take the exam

  Consider whether the string ends with null;

- For an INT_ A parameter of type RESULT must be cast to args ->args [i] as a long

  Long value:

```
long long int_ val;

int_ val = *((long long*) args->args[i]);
```

- For a REAL_ A parameter of type RESULT must be cast to args ->args [i] as a double

  Value:

```
double      real_ val;

real_ val = *((double*) args->args[i]);
```

- For a DECIMAL_ RESULT type parameters are processed in the same way as STRING_ RESULT One

  Sample.

unsigned long *lengths

- In the initialization function, the length array indicates the maximum string length for each parameter.
- For the main function call, length is the actual length of any string parameter of the currently being processed line

  Degrees.

- For INT_ RESULT or REAL_ For parameters of type RESULT, the length still contains the most

  Large length.

char *maybe_ null

In the initialization function, may_ Null indicates whether each parameter is allowed to be null.

explain

1 means allowed, 0 means not allowed.

char **attributes

The alias for each parameter, if there is no alias, is the actual name of the parameter, as shown in the following example:

select 1 from t1 group by udf_ func(1+2); Then args ->attributes [0] is "1+2", select 1+2 as plus from t1 group by udf_ func(plus); Then args ->attributes [0] is' plus'.

unsigned long *attribute_ lengths

The length of each attribute.

5.5.1.5 UDF return values and error handling

5.5.1.5.1 error handling

- If there are no errors, the initialization function should return 0, otherwise it should return 1;
- If an error occurs, func_ Init() should store error information in the message parameter and return it to the customer

  Household;
- The error message buffer is GBASE_ ERRMSG_ SIZE (currently in GBase, this length is 512

  Characters long, the buffer length should not be set too large, generally not exceeding 80 characters.

5.5.1.5.2 Function return value

- For long long and double functions, the return value of the main function func() is the function return value.
- For string functions, strings can be returned in the result and length parameters.
- Result is a buffer that is 255 bytes long. If the returned result does not exceed 255, it can be

One advantage of placing the results in the result is that there is no need to manage the memory of the result. For example:

```
memcpy(result, "result value", 12);

*length = 12;

return result;
```

- If the returned result exceeds 255 bytes, it needs to be processed in func_ Apply for space in init() or func() and place it in the

func_ Deinit() has been released, be careful not to cause memory leaks. For example, in Func_ In init:

At Func_ In init:

```
initid->ptr = (char *) malloc(MAX_LEN);
```

At Func_ In deinit:

```
free(initid->ptr);
```

- To indicate a NULL return value in the main function, set is_ Null to 1:

```
*is_ null = 1;
```

- To indicate an error return in the function, set the error parameter to 1:

```
*error = 1;
```

- If a certain line func() is set to * error of 1, then the current line function value is NULL, but it does not affect

The result of subsequent lines will be echoed, and func() will continue to be called.

5.5.1.6 Compile and create UDF

Step 1

Compile C or C++programs into shared libraries using the following command:

```
shell> gcc -fPIC func.c -shared -o func.so -I head_ file_ Path or

shell> gcc -fPIC func.cc -shared -o func.so -I head_ file_ Path or

shell> g++ -fPIC func.cc -shared -o func.so -I head_ file_ path
```

explain

head_ file_ Path is the storage path for the GBase header file used in func. c, usually located in the GBase installation directory

include/gbase。

Step 2

- Copy the compiled shared library (usually ending in. so, such as func. so above) to the plugin directory on the node in the cluster that initiated the creation of UDF functions.
- You can use the system variable plugin_ Dir gets the plugin directory, show variables like 'plugin'_

  dir'。

be careful

Some systems only recognize. so files starting with lib, and in this case, it is necessary to rename. so, such as func.so to libfunc.so

Step 3

After the shared library is copied to the plugin directory of the current node, the desired function can be created. The command is as follows:

```
gbase> CREATE FUNCTION    func    RETURNS STRING SONAME 'func.so';
```

You can use DROP JUNCTION to delete the function, as follows:

```
gbase> DROP FUNCTION func;
```

explain

- The CREATE FUSION and DROP FUSION statements update the system table func in the gbase database.

The function name, type, and shared library name are saved in this table. The current user must have insert and delete permissions to

Ability to create and delete functions;

- Cannot use CREATE JUNCTION to create a function that has already been created. If you need to recreate the function, you should delete it with DROP FUSION and then recreate it with CREATE FUSION.

  For example, if a new version of a function is recompiled in order for GBase to obtain the new version, the function needs to be deleted and recreated, otherwise GBase will continue to use the old version;

- When a user creates a UDF function, the current node will automatically copy the library file and send it to other nodes in the system;

- When the current node copies and sends UDF library files, if some nodes go offline abnormally and fail to copy UDF library files normally, after the abnormal node returns to normal, it will automatically pull the required UDF library files from the initiating node and complete the creation of UDF;

- The new function is loaded again every time the server starts, unless started with the -- skip grant tables option

GBase。 In this case, the initialization of the user-defined function is skipped, and the new function will become invalid.

Example

Copy the library file to the directory of the initiating node:

/opt/IP/gcluster/server/lib/gbase/plugin/ test_ udf.so

Initiate node creation udf:

drop function if exists substrb;

create function substrb returns string SONAME 'test_ udf.so';

5.5.1.7 UDF function serial parallel control

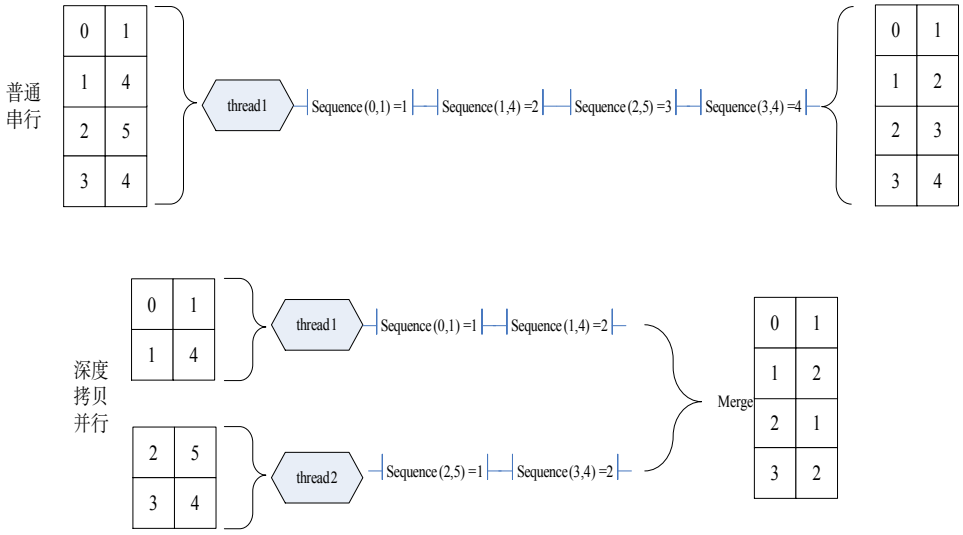
Expanding UDF supports multithreaded parallel computing, but with the following limitations:

- The serial/parallel execution of UDF functions requires the user to make their own judgments and pass in control parameters, and the database cannot make judgments on the serial/parallel execution of UDF functions.

- Non parallelizable UDF function: Firstly, the UDF function is a non deterministic function, even if UDF is passed in

  The same parameter, this UDF function will also obtain different values, and secondly, this UDF function is applicable to a set of numbers

  The result set obtained from set operations is also related to the order in which UDF functions call this set of data. For example, sequence, if parallel operations are performed, the following results will be obtained:



- GBase 8a MPP Cluster defaults to parallel execution for all UDF functions: when GBase_ parallel_ When the parallel switch 'execution' is turned on, the default execution mode for all UDF functions in gbased is also parallel. In this case, if the user needs to execute UDF functions serially, they need to change the UDF code themselves_ The configuration parameters passed to gbased for serial execution during INIT. Refer to the example below.

Example

The following is an example of a template that controls UDF serial execution:

```
//The user must provide my_ bool func_ init(UDF_INIT *initid, UDF_ARGS *args, char *message);

my_ bool func_ init (UDF_INIT *initid ,UDF_ARGS *args ,char *message )

{

    initid->extension = malloc(sizeof("no_parallel"));

     memcpy(initid->extension,"no_parallel",sizeof("no_parallel"));  .....
                                // no_ Parallel represents the serial execution of UDF functions

}



//User needs to provide void func_ deinit(UDF_INIT *initid)

void fullToHalf_ utf8_ deinit(UDF_INIT *initid)

{

     free(initid->extension);

     return;

}
```

5.5.1.8 UDAF Function Deployment and Usage

5.5.1.8.1 compile

Compile C or C++files into shared libraries using the following commands:

```
gcc -fPIC -Wall    func.c -shared -o func.so –I    head_ file_ path

g++ -fPIC -Wall    func.cc -shared -o func.so –I    head_ file_ path
```

explain

head_ file_ Path is the storage path for the GBase header file used in func. c, usually located in the GBase installation directory

include/gbase。

5.5.1.8.2 deploy

- Copy the compiled shared library (usually ending in. so, such as func. so above) to the plugin directory of the node in the cluster where the UDAF function is to be created. For example, copy the library file to the directory of the initiating node:/opt/IP/gcluster/server/lib/gbase/plugin.

- You can use the system variable plugin_ Dir gets the plugin directory, show variables like 'plugin'_ dir'。

be careful

Some systems only recognize. so files starting with lib, and in this case, it is necessary to rename. so, such as changing func. so to libfunc. so.

- After placing the shared library in the plugin directory, you can create UDAF functions.

5.5.1.8.3 establish

The syntax for creating UDAF functions is as follows:

```
CREATE AGGREGATE FUNCTION func RETURNS STRING SONAME 'func.so';
```

- Func is the function name of the UDAF created;
- STRING is the return type of the UDAF result, and currently supports returning string, int, real

  Four types of decimal;

- Func.so is the name of the shared library to load.

be careful

When creating a function, the capitalization of the function name and the function name in the. so file should be consistent, otherwise the creation will fail.

- When a user creates a UDAF function, the current node will automatically copy the library file and send it to the other nodes in the system.
- When the current node copies and sends UDAF library files, if some nodes go offline abnormally and fail to copy UDAF library files normally, after the abnormal node returns to normal, it will automatically pull the required UDAF library files from the initiating node and complete the creation of UDAF.

explain

The correspondence between SQL types and C/C++function return types is as follows:

Table -5195 Correspondence between SQL Types and C/C++Function Return Types

SQL Type	C/C++type
STRING	char *
INTEGER	long long
REAL	double
DECIMAL	char *

5.5.1.8.4 delete

After using UDAF, you can delete it. The syntax for deleting UDAF functions is as follows:

```
DROP FUNCTION    func;
```

explain

Func is the name of the UDAF to be deleted.

5.5.1.8.5 to update

- If a UDF/UDAF function needs to be updated, then the user needs to delete this UDF/UDAF and delete all UDF/UDAF functions of the same dynamic library (. so) referenced, and then recreate the UDF/UDAF function. Otherwise, there may be downtime when referencing later.
- We suggest building a dynamic library (. so) during the compilation of a UDF/UDAF function to avoid exceptions caused by UDF/UDAF updates.

- Cannot use CREATE AGGREGATE Function to create a function that has already been created.
- If you need to recreate the function, you should use DROP JUNCTION to delete it and then use CREATE

  Recreate AGGREGATE JUNCTION. For example, if a new

  Version in order for GBase to obtain a new version, the function needs to be deleted and recreated, otherwise GBase will continue

  Continue to use the old version.

5.5.1.8.6 **jurisdiction**

The CREATE AGGREGATE FUSION and DROP FUSION statements update the system table func in the GBASE 8a MPP Cluster database. The function name, type, and shared library name are saved in this table. The current user must have Insert and DELETE permissions to create and delete functions.

5.5.1.9 **UDAF Function Function Description**

5.5.1.9.1 **func_ init()**

Function prototype

```
my_ bool func_ init( UDF_INIT* initid, UDF_ARGS* args, char* message );
```

Function Description

- Check the number of parameters passed to func();
- Check if the parameter type is correct;
- If the parameter type is incorrect, in func_ Check in init and report an error, or in func_ add

  Self conversion in functions;
- Specify the maximum length of the returned result. For string/decimal, this value is the maximum byte of the return value

Number (decimal refers to the display of numerical values, including signs, decimal points, etc.). Return type is int

The value will be ignored when the real type is used;

- Specify the maximum decimal place for the returned result, mainly for decimal and real;
- Specify whether the result attribute does not contain NULL, or it can be NULL (column attribute in create table as select)

Sex will refer to this value.

Parameter Description

- Char * message if func_ If an error occurs in init, the user can copy the error information to the message,

Simultaneously, the function returns 1;

- This function is the initialization function of UDAF and is only called once at the beginning. It can be used for parameter validation,

Set output result attributes and other functions. If func_ Init() returns an error, the SQL statement returns an error message, and the function after UDAF will not be called.

5.5.1.9.2 func_ max_ buffer_ length ()

Function prototype

```
unsigned long long func_ max_ buffer_ length(UDF_INIT* initid, UDF_ARGS*
args,char * is_null,char * error);
```

Function Description

Set the maximum number of bytes required to cache intermediate results during group aggregation operations, which is the maximum memory required for a group.

- Parameter UDF_ The field type and maximum width of each parameter will be obtained in ARGS * args. Users can

Evaluate the maximum output width based on parameter characteristics;

- The program will assign an equal length buf to each group and reference it with initid ->ptr.

5.5.1.9.3 func_ clear ()

Function prototype

```
void func_ clear(UDF_INIT* initid, char* is_null, char* error);
```

Function Description

Reset the grouping buff (specified by the func_max_Buffer_length function). Users can set or clear the initial value of group buf through initid ->ptr.

5.5.1.9.4 func_ add ()

Function prototype

```
void func_ add(UDF_INIT* initid, UDF_ARGS* args,char * is_null, char * error);
```

Function Description

Within the same group, read the data from the current row and aggregate it onto the aggregation result, which is then stored on buf, which is referenced by initid ->ptr. This function has the following functions:

- When calling this function, buf stores the current aggregation result;
- When calling this function, args stores the column values that need to be aggregated for the current row;
- Perform corresponding aggregation operations based on the aggregation results stored in buf and the data that needs to be aggregated stored in args,

  And store the results back on buf;
- If the parameter type is incorrect, in func_ Check in init and report an error, or in func_ add

  Convert yourself in the function.

5.5.1.9.5 **func（）**

Function Description

The final aggregation result output of grouping, the difference in function return type and parameters depends on the type returned by the SQL function func() declared in the create aggregate function statement.

- The function that returns string/decimal in SQL has the following form:

```
char* func(UDF_INIT* initid, UDF_ARGS *args, char *result, unsigned long *length, uchar * is_null,uchar * error);
```

explain

When the UDAF returns a result type of string/decimal, this function is called to read and return the aggregation of each group

result. This function has the following functions:

- Read func_ The add function is ultimately stored in the aggregation result of initid ->ptr pointing to buf;
- Return the read aggregation results, and each group will be called once;
- is_ Null indicates whether the returned string is null;
- Error indicates whether the function has encountered an error;
- Length identifies the length (in bytes) of the returned string;
- When the field is of type date, the string type is returned.

- Return the integer function in SQL, as follows:

```
longlong func（UDF_INIT* initid , UDF_ARGS* args,uchar* is_null, uchar* error);
```

explain

This function is called when the UDAF returns a result type of long, reading and returning the aggregated results for each group.

- Return the real function in SQL, as follows:

```
double func(UDF_INIT *initid, UDF_ARGS* args,uchar* is_null, uchar* error);
```

Parameter Description:

- is_ Null indicates whether the return is NULL, and if the function is set to UDF during init_ INIT,

  maybe_ If null is 0 but returns NULL, SQL will report an error and exit.

- Does error return an error? If it is non zero, SQL will directly return the error.

explain

- The error description is roughly: UDAF funcname execute error, err_ no:error。
- is_ Although null and error are output parameters for all UDAF function interfaces, they are only present in some function interfaces

  It has practical significance. The following table shows:

Table -5196 Parameter Description

Output parameters	func_ max_ buffer_ length	func_ clear	func_ add	func
Is_ null	Not effective	Not effective	Not effective	take effect
error	Not effective	take effect	take effect	take effect

5.5.1.9.6 func_ deinit()

The end function of the UDAF is called only once after all lines are ended. It can be used to release the memory allocated by the initialization function.

be careful

All functions must be thread safe. It is not allowed to change globally shared or static variables in a function.

5.5.1.10 UDAF interface parameter description

5.5.1.10.1 UDF_ INIT structure

- This parameter is passed to all UDAF functions, pointing to a UDF_ INIT structure, used in the letter

  Transferring information between numbers;

- UDF_ The members of the INIT structure are listed below. The initialization function should initialize any function it wants to change

  Member, uninitialized member variables use default values.

```
typedef struct st_ udf_ init
{
   my_ bool maybe_ null;
   unsigned int decimals;
   unsigned long max_ length;
   char *ptr;
   my_ bool const_ item;
   void *extension;
   unsigned long *arg_ max_ lengths;
   unsigned long max_ buffer_ length;
} UDF_ INIT;
```

unsigned int decimals

Specify the maximum number of decimal places for the aggregation result, which is applicable to funs that return real or decimal types.

unsigned int max_ length

- Specify the maximum length of the returned result for DML such as create table like/as select;
- For functions that return string/decimal types, if the function output exceeds max_ Length, program behavior

  See Table 5-3. This table lists the default behavior of the express engine;
- max_ The configuration rules for length and decimals when UDAF returns different data types are as follows:

  See the table below:

Table -5197 Meaning of UDAF Return Value Types

Return Data Type	Type Meaning	Data result representation	
		max_ length	decimals
string	Varchar, date class	Maximum number of bytes (result exceeds error)	Not effective
decimal	decimal	Maximum Bytes (Result Overflow Error)	Maximum number of digits (result exceeds truncation)
real	double	(Using default) Not effective	Maximum number of digits (result exceeds truncation)
integer	bigint	(Using default) Not effective	Not effective

- For udf and udaf, for max in the init function_ The length and decimals values are checked for legality after init. The specific rules are as follows:

Table -5198 Legality Check Rules

Return Data Type	UDF/UDAF	
String	Maximum value is 65535 The minimum value is 0, and an error will be reported if it is illegal	Not effective
Decimal	The maximum value is 67 (including decimal parts, the maximum value is 67, excluding decimal parts, the maximum value is 66). After converting to precision, the maximum value is 65 Illegal reporting error	The maximum value is 30, reporting an error if it is illegal
Real	Not effective	The maximum value is 31, reporting an error if it is illegal
Integer	Not effective	Not effective

- In the above table, except for non effective cases, all other cases require init in UDF/UDAF

Display the setting max in the function_ The value of length or decimals; If not in the init function

The max set inside_ Length or decimals will be evaluated based on the parameters of the function to obtain a

Evaluated max_ The values of length or decimals are shown in the following table:

Table 5199 Func function returns a length exceeding max_ Processing behavior after length (select as an example)

Engine Category	String	Decimal
Express Engine	Exception thrown after exceeding	Exception thrown after exceeding

char *ptr

Point to the intermediate result buf assigned by the program for each group.

unsigned long *arg_ max_ lengths

The maximum byte length occupied by a parameter is an array, with each parameter occupying one.

unsigned long max_ buffer_ length

The maximum byte width of the buf that stores the intermediate results of the aggregation, which users can use in func_ max_ buffer_ Update and return this value in the length function.

my_ bool maybe_ null

If func does not return null, it is set to 0. If subsequent func returns a null value, SQL will report an error and exit, defaulting to 1.

my_ bool const_ item

If func always returns the same value, it is set to 1.

void * extention

The UDAF function does not use this field.

5.5.1.10.2 UDF_ ARGS structure

Used in func_ init、 func_ max_ buffer_ Length and Func_ In the add function, the structure is as follows:

```
typedef struct st_ udf_ args

{

   unsigned int arg_ count;.....


enum Item_ result *arg_ type;  .....

   char **args;            .....

                           .....

   unsigned long *lengths;.....


   char *maybe_ null;  .....


   char **attributes;

   unsigned long *attribute_ lengths;

   void *extension;

enum_ field_ types *field_ type

} UDF_ ARGS;
```

unsigned int arg_ count

Number of parameters. If the function has a fixed number of parameters, in func_ Check this value in the init initialization function. For example:

```
if (args->arg_count != 1)

{

        strcpy(message," func() requires one arguments");

        return 1;

}
```

enum Item_ result *arg_ type

The return type of each parameter. The possible type value is STRING_ RESULT, INT_ RESULT and REAL_ RESULT, DECIMAL_ RESULT。 Ensure that the parameter is of the required type, if not, return an error in func_ Check arg in init initialization function_ Type array to determine whether the parameter type meets the requirements. For example:

```
if (args->arg_type[0] != STRING_RESULT && args->arg_type[1] != INT_RESULT)
{
        strcpy(message," func() requires a string and an integer");
        return 1;
}
```

char **args

Args buffer the parameter values of the function to func_ Init() and func_ add()。

enum enum_ field_ types *field_ type

The type of each field in the parameter.

The above three fields describe the parameters. For different fields, the program returns them to the user function according to the following default behavior. It is recommended that the user function be processed according to the following table.

Table 5200 field_ type、arg_ Type return description

field_ type	arg_ type	User function processing method (Storage format and user handling in args)
String(varchar,char) The corresponding fields are (GBASE_TYPE_VARCHAR, GBASE_TYPE_STRING)	STRING_ RESULT	Return Type: String Char * tmp tmp = args->args[i]
Decimal (GBASE_TYPE_NEWDECIMAL)	DECIMAL_ RESULT	Return Type: String Char * tmp tmp = args->args[i]
Date (time,timestamp,date…) The corresponding fields are (GBASE_TYPE_TIMESTAMP, GBASE_TYPE_TIME,	STRING_ RESULT	Return type: ISO standard fixed length time string, with specific lengths for the following types. Date 10 bytes, such as' 2017 03 12 '

GBASE_TYPE_DATE,...)		The character lengths returned by the two types of datetime and timestamp may be 19 or 26, as shown in the following examples: '2017 03 12 23:12:56', '2017 01 01 01 01:02:03.456789', and '2017 01 01 01 01:02:03.010000' (the rule is that microseconds are zero and do not return microseconds, otherwise a 6-bit fixed length microsecond is returned) The length of the string returned by the Time data type may be 10 or 17, as shown in the following examples: '23:12:56' '01:02:03.456789' '01:02:03.010000' '-111:02:03.0110000' (rule shows negative signs, omits positive signs, and hours can be up to three digits) Parsing method: char *tmp tmp = args->args[i]
The fields corresponding to Int (short, long, tiny...) are (GBASE_TYPE_SHORT, GBASE_TYPE_LONG, GBASE_TYPE_TINY,...)	INT_ RESULT	Return type: 8 byte space Parsing method: Longlong int_ val; Int_ val = *(longlong *)args->args[i]
Double(float,double) The corresponding fields are (GBASE_TYPE_FLOAT, GBASE_TYPE_DOUBLE)	REAL_ RESULT	Return type: 8-byte space, stored as double. double real_ val real_ val = *((double*) args->args[i]);

unsigned long *lengths

- For func_ Add() call, where lengths are any string parameters currently being processed for the line

  The actual length of (STRING_RESULT, DECIMAL_RESULT);

- For INT_ RESULT or REAL_ For parameters of type RESULT, the length still contains the most

  Large length, usually 8 bytes in length.

char *maybe_ null

In the initialization function, may_ Null indicates whether each parameter may be null (1 indicates possible, 0 indicates impossible). This variable is generally determined by the field attribute of the incoming parameter, and should not be specified by the user.

char **attributes

The alias for each parameter, if no alias exists, is the actual name of the parameter, such as:

select 1 from t1 group by udf_ func(1+2);

Then **args ->attributes [0] is "1+2".**

select 1+2 as plus from t1 group by udf_ func(plus);

Then **args ->attributes [0] is' plus'.**

unsigned long *attribute_ lengths

The length of each attribute.

5.5.1.11 Example of using UDAF function

After creating the UDAF, use the UDAF function just like using regular built-in functions.

The following example uses the UDAF function newest to find the field quantity value of the row where the maximum value of field n1 is located.

```
drop function newest;
CREATE AGGREGATE FUNCTION newest RETURNS STRING SONAME 'newest.so';
drop table if exists t;
create table t(n1 date,quantity    varchar(10));
insert into t values('2011-01-01','aa');
insert into t values('2012-02-01','bb');
insert into t values('2012-01-02','cc');


gbase> select newest(n1,quantity) from t;
+----------------------+
```

```
| newest(n1,quantity) |
+---------------------+
| bb                  |
+---------------------+
1 row in set (Elapsed: 00:00:00.01)
```

5.5.1.12 Python UDF

In GBase 8a MPP Cluster, the PL/Python stored procedure language is used to support UDF functions written in Python. PL/Python exists in the form of an 'untrusted' language, which means it does not limit how users can use Python. So this language is named plpythonu. If Python provides new security mechanisms in the future, it will provide the PLPython language. Untrusted PL/Python writers must be cautious in writing these functions and not use them for illegal operations, as this feature allows users with DBA identity to execute arbitrary scripts. Only super users have permission to create these functions.

5.5.1.12.1 Anonymously execute any Python script

Support parameter input and output parameters to return execution results of string type.

Grammar format

```
python(code,[<parameter_1>[,…] [,parameter_n])
```

explain

- Code is a Python code of string type;
- Parameter is optional, and type is the data type supported by GBase 8a MPP Cluster.

Example

```
gbase> select python('return 1+1');
+----------------------+
| python('return 1+1') |
+----------------------+
| 2                    |
```

```
+----------------------+

gbase> select python('import datetime\nreturn datetime.datetime.now()');

+------------------------------------------------------------+

| python('import datetime\nreturn datetime.datetime.now()') |

+------------------------------------------------------------+

| 2017-01-03 17:38:47.138760                                 |

+------------------------------------------------------------+

gbase> select python('import platform\nreturn platform.platform()') as result;

+------------------------------------------------------------+

| result                                                     |

+------------------------------------------------------------+

| Linux-2.6.32-431.el6.x86_ 64-x86_ 64-with-redhat-6.5-Santiago |

gbase> set @a=1,@b=2;

gbase> select python('return a+b',@a,@b);

+----------------------------+

| python('return a+b',@a,@b) |

+----------------------------+

| 3                          |

+----------------------------+
```

5.5.1.12.2 Python custom function support

Grammar format

```
CREATE FUNCTION <func_ name>([<parameter_1>[,…] [,parameter_n]])
RETURNS    type
$$
   <Python Function Definition>
$$ LANGUAGE plpythonu
```

Parameter Description

- <func_ Name>The name of the function to be created. The names of functions must be unique within the same database.

Function names only allow a-z, A-Z, 0-9, and underscores, and cannot contain only numbers;

- ([<parameter_1>[,...] [, parameter_n]]) Define the parameters of a function, with each parameter

The definition format is:<parameter name><parameter data type>.

explain

- The string delimiter '$$' has been added here, which can avoid using escape in Python function definitions compared to the delimiter '';
- <Parameter name>must be unique within the same function, only a-z, A-Z, 0-9, and underscores are allowed, and cannot only contain numbers;
- <Parameter Data Type>Specify the data type of the parameter;
- <Function Definition>is a combination of a series of Python statements that contain data operations to complete certain functional logic;
- When defining a function, the parentheses after the function name are required and cannot be omitted even if there are no parameters;
- Type is the data type supported by GBase 8a MPP Cluster.

Example

Example 1

```
gbase> create function getUrlTitle(url varchar) returns varchar $$
            import lxml.html,urllib
          return lxml.html.parse(urllib.urlopen(url)).xpath("//title")[0].text
    $$ LANGUAGE plpythonu;
gbase> select getUrlTitle(' http://192.168.6.253/ ') as result;
+----------------------+
| result               |
+----------------------+
| Login to Phabricator |
+----------------------------------+
gbase> drop function getUrlTitle;
```

Example 2

Tinyint type support, examples are as follows:

```
gbase> create function type_ tinyint(i tinyint) returns tinyint $$ return i $$ LANGUAGE plpythonu;
gbase> select type_ tinyint(127);
+-------------------+
| type_ tinyint(127) |
+-------------------+
|                 127 |
+-------------------+
```

Example 3

Smallint type support, examples are as follows:

```
gbase> create function type_ smallint(i smallint) returns smallint $$ return i $$ LANGUAGE plpythonu;
gbase> select type_ smallint(32767);
+----------------------+
| type_ smallint(32767) |
+----------------------+
|                 32767 |
+----------------------+
```

Example 4

Support for int type, examples are as follows:

```
gbase> create function type_ int(i int) returns int $$ return i $$ LANGUAGE plpythonu;
gbase> select type_ int(2147483647);
+----------------------+
| type_ int(2147483647) |
+----------------------+
|           2147483647 |
```

```
+----------------------+
```

Example 5

Bigint type support, examples are as follows:

```
gbase> create function type_ bigint(i bigint) returns bigint $$ return i $$ LANGUAGE pl
pythonu;
gbase> select type_ bigint(9223372036854775806);
+----------------------------------+
| type_ bigint(9223372036854775806) |
+----------------------------------+
|                  9223372036854775806 |
+----------------------------------+
```

Example 6

Float type support, examples are as follows:

```
gbase> create function type_ float(i float) returns float $$ return i $$ LANGUAGE plpyth
onu;
gbase> select type_ float(3.40E+38);
+----------------------+
| type_ float(3.40E+38) |
+----------------------+
| 3.39999995214436e+38 |
+----------------------+
```

Example 7

Double type support, examples are as follows:

```
gbase> create function type_ double(i double) returns double $$ return i $$ LANGUAGE
 plpythonu;
gbase> select type_ double(1.7976931348623157E+308);
+-------------------------------------+
```

```
| type_ double(1.7976931348623157E+308) |
+---------------------------------------+
|                    1.79769313486232e+308 |
+---------------------------------------+
```

Example 8

Varchar type support, examples are as follows:

```
gbase> create function type_ varchar(i varchar) returns varchar $$ return i $$ LANGUAGE plpythonu;
gbase> select type_ varchar('abc');
+---------------------+
| type_ varchar('abc') |
+---------------------+
| abc                  |
+---------------------+
```

Example 9

SQL NULL and Python None conversion support, as shown in the following example:

```
gbase> select type_ varchar(NULL);
+--------------------+
| type_ varchar(NULL) |
+--------------------+
| NULL                |
+--------------------+
gbase> create function type_ none() returns varchar $$ return None $$ LANGUAGE plpythonu;
gbase> select type_ none();
+-------------+
| type_ none() |
+-------------+
```

```
| NULL          |
+-------------+
```

5.5.1.12.3 Constraints and limitations

The data input parameter data type mapping relationships supported by Python UDF in GCluster 8a MPP Cluster are shown in the following table:

Table 5201 Data Input Parameter Data Type Mapping Relationships

GBase	Python2
TINYINT/SMALLINT/INT/BIGINT	long
FLOAT/DOUBLE	float
VARCHAR	Str (database encoding)
NULL	None

GCluster 8a MPP Cluster's Python UDF supported data return value data type mapping relationship:

Table 5202 Return Value Data Type Mapping Relationships

GBase	Python2
TINYINT/SMALLINT/INT/BIGINT	long
FLOAT/DOUBLE	float
VARCHAR	Str (database encoding)
NULL	None

explain

Unsupported features include:

- Sharing variables between Python UDF functions is not supported;
- Python UDF is not supported as a trigger;
- Python syntax check is not supported. In case of syntax error, the custom function can be created successfully. If there is a syntax error,

  Clear error messages can be prompted during execution;
- The data type only supports GBase data types in the list, and does not support DECIMAL, CHAR, or TEXT classes

  Type;
- The parameter list does not support OUT type definition and does not support returning values from the parameter list.

5.5.2 GBMLLib (Data Mining Module)

5.5.2.1 Introduction to GBMLLib

5.5.2.1.1 Introduction to GBMLLib

GBMLLib is a data mining and machine learning extension library for GBase 8a MPP Cluster, added as a plugin to GBase 8a MPP Cluster. Through its provided machine learning algorithms, GBase 8a MPP Cluster can conduct in-depth analysis and mining of user data, transforming user data into user value.

GBMLLib provides SQL based machine learning algorithms, which currently include regression algorithms (linear regression), classification algorithms (logistic regression, support vector machines), and clustering algorithms (K-Means). It also provides some basic functions for array operations and linear algebra calculation.

5.5.2.1.2 Technical characteristics

GBMLLib has the following technical features:

- SQL interface: GBMLLib provides SQL based data mining algorithms, and the training, evaluation, and prediction of models are executed through SQL statements, making it very easy for data analysts to master and combine with their existing skills to fully unleash their creativity and improve work efficiency.
- In database analysis: Unlike other analysis tools that require moving data from the database to the analysis node for processing through APIs or ODBC, GBMLLib's analysis algorithm runs in the form of a database udf/udaf within the threads of GBase8a, and is scheduled through the execution plan of GBase8a to minimize data movement and improve running speed.
- Convenient expansion: GBMLLib is added to GBase8a in the form of plug-in, and adopts flexible software architecture to facilitate the subsequent addition of new data mining and machine learning algorithms.

5.5.2.1.3 Product Function Introduction

The data mining functions supported by GBMLLib are shown in the table below:

Table -5203 Data Mining Functions Supported by GBMLLib

Category	Algorithm	Description
regression	linear regression	A statistical analysis method that uses regression analysis in mathematical statistics to determine the quantitative relationship of interdependence between two or more variables
classification	Logistic regression	A second class classification model that establishes regression formulas for classification boundaries based on existing data, in order to classify them
	Support Vector Machine (SVM)	The second class classification model is defined as the Linear classifier with the largest interval in the feature space, and the learning strategy is to maximize the interval, which can finally be transformed into the solution of a convex Quadratic programming problem.
clustering	K-Means clustering	Divide n points (which can be a single observation or instance of a sample) into k clusters, so that each point belongs to the cluster corresponding to its nearest mean (i.e. cluster center), and use it as the clustering criterion.

5.5.2.1.4 Explanation of Terms

Characteristic attributes

In relational database, it refers to the table fields involved in data mining, which can also be called characteristic attributes, characteristic fields, characteristic factors, characteristic variables, etc.

Independent Variable

Refers to characteristic indicators that can function independently and are generally immutable, such as the user's age, gender, occupation, and so on. Also known as driving factors, driving variables, dependent variables, etc. Big data mining generally discovers the variation patterns of dependent variables through the changes in different values of independent variables.

Dependent Variable

Refers to characteristic attributes that vary depending on one or more independent variables. It can also be called driven factor, dependent variable, target variable, categorical variable, etc. The dependent variable usually manifests as the behavior pattern of an entity, such as whether to purchase a certain product, whether it has the ability to repay a loan, and so on. Big data mining generally focuses on discovering the

changes in dependent variables.

Data Mining Algorithm:

A data manipulation process described in mathematical language using mathematical methods, also known as machine learning algorithms.

5.5.2.2 GBMLLib installation

system requirements

Hardware platform (no special requirements, refer to unified specifications).

Software platform: RedHat7 or above or Suse12 or above

Installation steps

Please refer to Chapter 3 for software installation steps for cluster installation. The following assumes that the cluster has been installed:

Execute the following command on any management node in the cluster:

```
# export GBASE="$GCLUSTER_HOME/bin/gccli -uroot"
# $GCLUSTER_ HOME/bin/gbase_ install_ mllib

Installing gbmllib plugin...

Loading from file: /opt/gcluster/server/lib/gbase/plugin/mllib/regress/linear.sql
Loading from file: /opt/gcluster/server/lib/gbase/plugin/mllib/regress/logreg.sql
Loading from file: /opt/gcluster/server/lib/gbase/plugin/mllib/utilities/utilities.sql
Loading from file: /opt/gcluster/server/lib/gbase/plugin/mllib/array/array_ func.sql
Loading from file: /opt/gcluster/server/lib/gbase/plugin/mllib/kmeans/kmeans.sql
Loading from file: /opt/gcluster/server/lib/gbase/plugin/mllib/recursive_
partitioning/decision_ tree.sql
Loading from file: /opt/gcluster/server/lib/gbase/plugin/mllib/stats/correlation.sql
Loading from file: /opt/gcluster/server/lib/gbase/plugin/mllib/sample/sample.sql
Loading from file: /opt/gcluster/server/lib/gbase/plugin/mllib/svm/svm.sql
Loading from file: /opt/gcluster/server/lib/gbase/plugin/mllib/linalg/matrix_
ops.sql
```

```
Loading from file:
/opt/gcluster/server/lib/gbase/plugin/mllib/linalg/linalg.sql
```

The installation is now complete.

5.5.2.3 Usage Overview

GBMLLib is provided in the form of a GBase 8a MPP CLuster database plugin, providing a series of SQL functions for data mining and machine learning. Data analysts can perform machine learning modeling, model evaluation, and data prediction by executing SQL commands on the GBase8a client.

5.5.2.3.1 Data mining process

technological process

- Prepare data:
  GBMLLib directly analyzes and mines the data tables of GBase 8a MPP CLuster. During the data preparation stage, users organize the data to be mined into data tables and store them in GBase 8a MPP CLuster.
- Call the GBMLLib mining method to mine the training data, and the mining results are stored in the result table.
  The mining functions of GBMLLib basically follow the naming rules of train predict. For a mining algorithm, algorithms will be provided_ The training function and algorithm of train()_ The prediction function of predict() is saved in the result table by training the model through the train function; Use the model saved in the result table to make predictions using the predict function.
- The following statement describes the training function call method for logistic regression, which will be explained in detail in the following chapters:

```
SELECT mllib.logregr_ train(
        'test.patients',                    -- source table
        'test.patients_ logregr',           -- result table
        'second_ attack',                   -- dependent variable
        'array double[1, treatment, trait_anxiety]',
-- independent variable
```

```
        20,                                    -- max_ iter
        'cg',                                   -- optimizer
        0.0001                                 -- tolerance
);
```

- Query the result table, view the mining results, and save the training results in the user specified result table (such as patients_logregr in the example above). At the same time, a summary table will be generated to record the training summary information. The table name of the summary table is the result table plus "_summary". As shown in the example above, patients will be generated_ logregr_ Summary table for the summary. You can view mining results and model information through the query result table and summary table.
- Call the prediction function to predict new data.
- The prediction of new data is completed through the predict() function, which typically takes the model coefficients saved in the result table and the dependent variables of the new data table as inputs. The prediction statement for logistic regression is as follows:

```
SELECT p.id,
    mllib.logregr_ predict(
        coef,                                                -- coefficients
        array double[1, treatment, trait_anxiety] -- independent variable
  ) as predict,
    p.second_ attack
FROM patients p, patients_ logregr m
ORDER BY p.id;
```

5.5.2.3.2 Usage Instructions for Array Types

The data mining algorithm provided by GBMLLib involves a large number of linear algebra operations and needs to deal with vectors and matrices. Vector and matrix are stored in GBase 8a MPP Cluster through BLOB type, and functions are provided to assemble integer and floating point data into arrays and display array contents.

Assemble an array

grammar

ARRAY type[expr1 [, expr2 …]]

Type specifies the type of data to save to the array. Currently supports double and bigint.

Example

- Create a t1 table and insert data of array type.

```
gbase> create table t1(a int, b blob);
Query OK, 0 rows affected (Elapsed: 00:00:00.01)

gbase> insert into t1 values(1, ARRAY BIGINT[1,2]);
Query OK, 1 row affected (Elapsed: 00:00:00.01)

gbase> insert into t1 values(2, ARRAY BIGINT[3,4]);
Query OK, 1 row affected (Elapsed: 00:00:00.00)
```

Display data in an array

grammar

ARRAY_ TEXT(expr)

Example

- Display the contents of the array in the t1 table.

```
gbase> select a, ARRAY_ TEXT(b) from t1;
+------+---------------+
| a    | ARRAY_ TEXT(b) |
+------+---------------+
|    1 | {1,2}          |
|    2 | {3,4}          |
+------+---------------+
2 rows in set (Elapsed: 00:00:00.00)
```

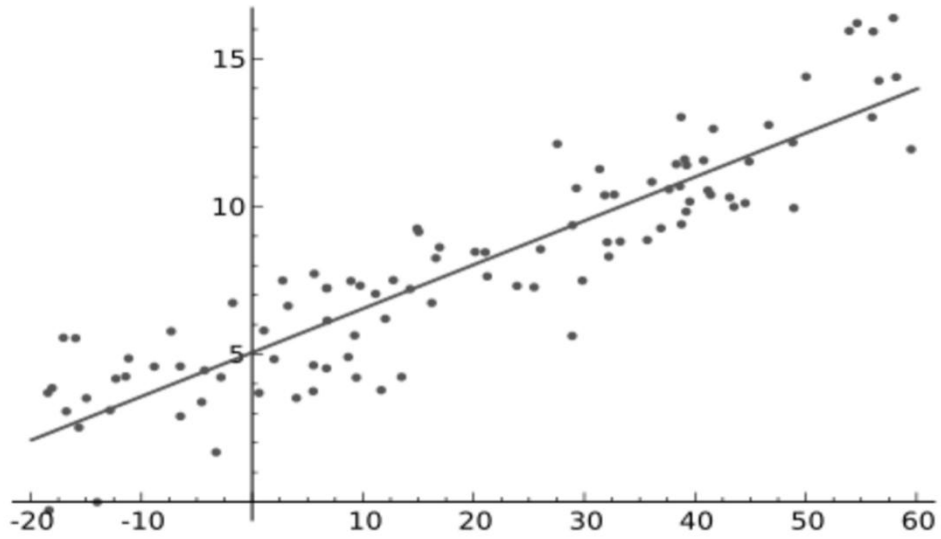
You can refer to the examples of various mining algorithms below for more information on the use of array types.

5.5.2.4 Regression algorithm

5.5.2.4.1 linear regression

A linear regression model is used to discover the linear relationship between a pure dependent variable and one or more independent variables, and to predict the results using this regression formula.

Figure -513 Schematic diagram of linear regression



The schematic diagram of linear regression is shown in the figure above. Through training, a set of coefficients is found to achieve the best fit to the data, and the coefficients are saved in the result table for prediction of new data.

5.5.2.4.1.1Training function

grammar

The syntax of the training function for linear regression is as follows:

```
linregr_ train( source_table,
                out_ table,
                dependent_ varname,
                    independent_ varname)
```

Parameter Description

- source_ Table: Input table containing training data.

- out_ Table: Save the result table of the training results.
- dependent_ Varname: The column name of the dependent variable.
- independent_ Varname: The column name and array type of the argument.

Result Table Explanation

After the training function is successfully executed, a result table will be created to save the model information, which includes the following fields to represent the model information:

Coef: Correlation coefficient, used for prediction.

5.5.2.4.1.2Predictive function

grammar

The syntax of the prediction function for Linear regression is as follows:

```
linregr_ predict(coefficients,
                                    independent_ varname
                            )
```

Parameter Description

- Coefficients: The correlation coefficients saved in the model result table.
- independent_ Varname: The column name and array type of the argument.

5.5.2.4.1.3Example

- Create a data table for training and insert training data.

```
CREATE TABLE houses (id INT, tax INT, bedroom INT, bath double, price INT,
            size INT, lot INT);
INSERT INTO houses VALUES
      ( 1,   590,          2,      1,   50000,    770, 22100),
      ( 2, 1050,          3,      2,   85000, 1410, 12000),
      ( 3,     20,          3,      1,   22500, 1060,    3500),
      ( 4,   870,          2,      2,   90000, 1300, 17500),
      ( 5, 1320,          3,      2, 133000, 1500, 30000),
      ( 6, 1350,          2,      1,   90500,    820, 25700),
      ( 7, 2790,          3,    2.5, 260000, 2130, 25000),
```

```
       ( 8,    680,            2,       1, 142500, 1170, 22000),
       ( 9, 1840,              3,       2, 160000, 1500, 19000),
       (10, 3680,              4,       2, 240000, 2790, 20000),
       (11, 1660,              3,       1,    87000, 1030, 17500),
       (12, 1620,              3,       2, 118600, 1250, 20000),
       (13, 3100,              3,       2, 140000, 1760, 38000),
(14, 2070,             2,       3, 148000, 1550, 14000),
(15,    650,            3,    1.5,    65000, 1450, 12000);
```

- Train a classification model.

```
SELECT mllib.linregr_ train( 'madtest.houses',
              'madtest.houses_ linregr',
              'price',
              'ARRAY DOUBLE [1, tax, bath, size]'
       );
```

- View training results.

```
gbase> SELECT array_ text(coef) FROM houses_ linregr \G;
*************************** 1. row ***************************
array_ text(coef):
{-12849.4168959872,28.9613922651765,10181.6290712648,50.516894915354}
1 row in set (Elapsed: 00:00:00.00)
```

- Use a model to predict and display the difference between the actual and predicted values.

```
gbase> SELECT
houses.id as id,
houses.price as price,
mllib.linregr_ predict(
ARRAY DOUBLE [1,tax,bath,size],
m.coef
) as predict,
       price - mllib.linregr_ predict(
ARRAY DOUBLE [1,tax,bath,size],
              m.coef
       ) as residual
```

```
FROM houses, houses_ linregr m;



+------+--------+------------------+-------------------+
| id     | price    | predict                  | residual                  |
+------+--------+------------------+-------------------+
|       1 |    50000 | 53317.4426965542 | -3317.44269655424 |
|       2 |    85000 | 109152.124955627 | -24152.1249556268 |
|       3 |    22500 | 51459.3486308563 | -28959.3486308563 |
|       4 |    90000 |    98382.215907206 | -8382.21590720605 |
|       5 | 133000 | 121518.221409606 |    11481.7785903937 |
|       6 |    90500 | 77853.9455638561 |    12646.0544361439 |
|       7 | 260000 | 201007.926371721 |    58992.0736282788 |
|       8 | 142500 | 76130.7259665617 |    66369.2740334383 |
|       9 | 160000 | 136578.145387498 |    23421.8546125019 |
|      10 | 240000 |    255033.90159623 | -15033.9015962295 |
|      11 |    87000 | 97440.5250982852 | -10440.5250982852 |
|      12 | 118600 | 117577.415360321 |    1022.58463967926 |
|      13 | 140000 | 186203.892319613 | -46203.8923196126 |
|      14 | 148000 | 155946.739425521 | -7946.73942552117 |
|      15 |    65000 | 94497.4293105379 | -29497.4293105379 |
+------+--------+------------------+-------------------+
15 rows in set (Elapsed: 00:00:00.00)
```

The above results are linear regression predictions of housing prices, where the price column represents the actual price of the house, the predict column represents the price predicted by the model, and the residual column represents the difference between the actual price and the predicted price. From the results, it can be seen that the linear model is basically correct.

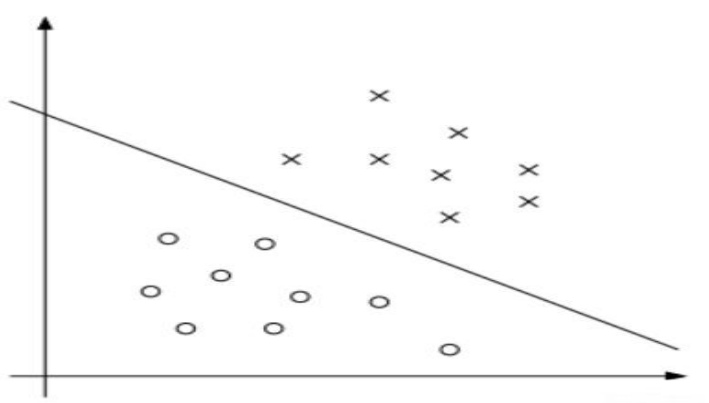
5.5.2.5 **Classification algorithm**

5.5.2.5.1 **Logistic regression**

The binary logistic regression model is used to discover the relationship between a binary (value 0 or 1) dependent variable and an independent variable. By establishing a

regression formula, the classification boundary of the independent variable data is determined, and classification predictions are made using this regression formula. Due to the use of logistic function as the discriminant classification function, it is called logistic regression.

Figure -514 Schematic diagram of logistic regression



The schematic diagram of logistic regression is shown in the above figure. Through training, a segmented hyperplane is found, the binary (data marked with x and data marked with o) training data is distinguished, and the coefficient value of this segmented hyperplane is saved in the result table for prediction of new data.

5.5.2.5.1.1Training function

grammar

The syntax of the training function for logistic regression is as follows:

```
logregr_ train( source_table,
                out_ table,
                dependent_ varname,
                independent_ varname,
                max_ iter,
                optimizer,
                tolerance
              )
```

Parameter Description

- source_ Table: Input table containing training data.
- out_ Table: Save the result table of the training results.

- dependent_ Varname: The column name of the dependent variable. The dependent variable column should be a Boolean value, and non Boolean values will be implicitly converted to Boolean values during processing.
- independent_ Varname: The column name and array type of the argument.
- max_ Iter: The maximum number of iterations.
- Optimizer: The optimizer used during the iteration process.
- Tolerance: Tolerance. If the logarithmic likelihood difference between two iterations is less than this value, the iteration ends.

Result Table Explanation

After the training function is successfully executed, a result table will be created to save the model information, which includes the following fields to represent the model information:

- Coef: Correlation coefficient, used for prediction.
- log_ Likelihood: Logarithmic likelihood value, which evaluates the parameters of the model during training.
- std_ Err: The standard deviation of the correlation coefficient.
- z_ Stats: z-statistic of correlation coefficient.
- num_ rows_ Processed: The number of data rows processed.
- num_ missing_ rows_ Skipped: The number of data rows skipped.
- num_ Iterations: Number of iterations.

Summary Table Description

At the end of the training, a summary table will be generated, which is called the result table name plus "_ Summary ", the fields in the summary table are explained as follows:

- Method: The name of the mining algorithm, which is logregr.
- source_ Table: Enter a table name.
- out_ Table: The name of the result table.
- dependent_ Varname: The name of the dependent variable.
- independent_ Varname: The name of the independent variable.

- optimizer_ Params: Optimizer parameters, maximum number of iterations, tolerance, etc.
- num_ failed_ Groups: The number of groups that failed training.
- num_ rows_ Processed: The number of data rows processed.
- num_ missing_ rows_ Skipped: The number of data rows skipped.

5.5.2.5.1.2Predictive function

grammar

The syntax of the predictive function for logistic regression is as follows:

```
logregr_ predict(coefficients,
                 ndependent_ varname
                )
```

Parameter Description

- Coefficients: The correlation coefficients saved in the model result table;
- independent_ Varname: The column name and array type of the argument.

5.5.2.5.1.3Example

- Create a data table for training and insert training data.

```
CREATE TABLE patients( id INTEGER NOT NULL,
                       second_ attack INTEGER,
                            treatment INTEGER,
                       trait_ anxiety INTEGER);
INSERT INTO patients VALUES
( 1,    1,    1,    70),
( 3,    1,    1,    50),
( 5,    1,    0,    40),
( 7,    1,    0,    75),
( 9,    1,    0,    70),
(11,    0,    1,    65),
(13,    0,    1,    45),
(15,    0,    1,    40),
(17,    0,    0,    55),
```

```
(19,      0,      0,     50),
( 2,      1,      1,     80),
( 4,      1,      0,     60),
( 6,      1,      0,     65),
( 8,      1,      0,     80),
(10,      1,      0,     60),
(12,      0,      1,     50),
(14,      0,      1,     35),
(16,      0,      1,     50),
(18,      0,      0,     45),
(20,      0,      0,     60)
```

- Train a classification model.

```
SELECT mllib.logregr_ train(
            'test.patients',
            'test.patients_ logregr',
            'second_ attack',
            'array double[1, treatment, trait_anxiety]',

            20,
            'cg',
zero point zero zero zero one
 );
```

- View training results.

```
gbase> SELECT * FROM patients_ logregr\G
*************************** 1. row ***************************
                        coef: -5.828, -0.888858, 0.108851
            log_ likelihood: -9.70259
                    std_ err: 2.70859, 1.08267, 0.0461127
                    z_ stats: -2.15168, -0.820985, 2.36054
       num_ rows_ processed: 20
num_ missing_ rows_ skipped: 0
            num_ iterations: 17
1 row in set (Elapsed: 00:00:00.00)
```

```
gbase> select * from test.patients_ logregr_ summary\G
*************************** 1. row ***************************
                    method: logregr
                source_ table: test.patients
                   out_ table: test.patients_ logregr
         dependent_ varname: second_ attack
       independent_ varname: array double[1, treatment, trait_anxiety]
          optimizer_ params: optimizer=cg, max_ iter=20, tolerance=0.0001
             num_ all_ groups: 1
          num_ failed_ groups: 0
        num_ rows_ processed: 20
num_ missing_ rows_ skipped: 0
                grouping_ col: NULL
1 row in set (Elapsed: 00:00:00.00)
```

- Use a model for prediction.

```
gbase>   SELECT p.id,
                    mllib.logregr_ predict(
                        coef,
                        array double[1, treatment, trait_anxiety]
                    ) as predict,
                    p.second_ attack
         FROM patients p, patients_ logregr m
         ORDER BY p.id;
+----+---------+---------------+
| id | predict | second_ attack |
+----+---------+---------------+
|  1 | 1       |             1 |
|  2 | 1       |             1 |
|  3 | 0       |             1 |
|  4 | 1       |             1 |
|  5 | 0       |             1 |
|  6 | 1       |             1 |
|  7 | 1       |             1 |
|  8 | 1       |             1 |
|  9 | 1       |             1 |
```

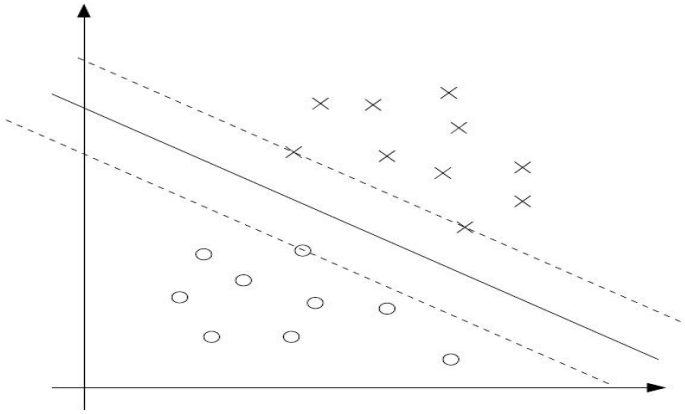
```
| 10 | 1          |              1 |
| 11 | 1          |              0 |
| 12 | 0          |              0 |
| 13 | 0          |              0 |
| 14 | 0          |              0 |
| 15 | 0          |              0 |
| 16 | 0          |              0 |
| 17 | 1          |              0 |
| 18 | 0          |              0 |
| 19 | 0          |              0 |
| 20 | 1          |              0 |
+----+----------+---------------+
20 rows in set (Elapsed: 00:00:00.00)
```

The id column output by the prediction statement represents different patients, and the value of the predict column represents the predicted value of whether the patient will relapse (0 represents no recurrence, 1 represents recurrence), second_ The value in the attachment column is the true value of whether the patient has relapsed, and comparing the two values indicates that using a logistic regression model for analysis is more appropriate.

5.5.2.5.2 Support Vector Machine

Support Vector Machine (SVM) is also a binary classification algorithm, and a significant advantage of the SVM model is its robustness to noisy data. Linear support vector machine uses linear hyperplane to classify data.

Figure -515 Vector machine algorithm



Support vector machine model is to solve the problem of maximizing the distance (called boundary) between the segmented hyperplane and the nearest training data. The points on the boundary line are called support vectors. When the support vector is determined, the changes in points that are not on the boundary line will not affect the results of the model, which ensures the robustness of the support vector machine.

5.5.2.5.2.1Training function

grammar

The syntax of the training function for the support vector machine classification model is as follows:

```
svm_ classification(
        source_ table,
        model_ table,
        dependent_ varname,
        independent_ varname,

        params
          )
```

Parameter Description

- source_ Table: Input table containing training data.
- model _ Table: Save the result table of the training results.
- dependent_ Varname: The column name of the dependent variable.
- independent_ Varname: The column name and array type of the argument.
- Params: Model training parameters.

  The model training parameters (params) are key value pairs separated by commas, and the supported key values are as follows:

  init_ Stepsize (initial step size)

  Default value: [0.01]. Initial learning step size. A relatively small value can ensure convergence results, while a larger value can improve cultivation speed.

  decay_ Factor (attenuation coefficient).

Default value: [0.9]. Control the learning step used during the iteration process: 0 represents a constant step size- 1 represents reverse scaling, step size=initial step size/number of iterations;> 0 represents exponential decay, step size=initial step size * decay coefficient ^ number of iterations.

max_ Iter (maximum number of iterations)

Default value: [100]

Tolerance

Default value: 1e-10. When the difference between two iterations of the model is less than the tolerance, the iteration ends.

Lambda (regularization parameter)

Default value: [0.01]. must be greater than 0 and cannot be negative.

Result Table Explanation

After the training function is successfully executed, a result table will be created to save the model information, which includes the following fields to represent the model information:

- Coef: Correlation coefficient, used for prediction.
- Loss: loss function value.
- norm_ of_ Gradient: the gradient value of the loss function.
- num_ Iterations: Number of iterations.
- num_ rows_ Processed: The number of data rows processed.
- num_ rows_ Skipped: The number of data rows skipped.
- dep_ var_ Mapping: Possible values of the dependent variable.

Summary Table Description

After the training, a summary table will be generated, which is called the result table name plus "_ Summary ", the fields in the summary table are explained as follows:

- Method: The name of the mining algorithm, where SVC represents the SVM classification algorithm.
- source_ Table: Enter a table name.
- model_ Table: The name of the result table.

- dependent_ Varname: The name of the dependent variable.
- independent_ Varname: The name of the independent variable.
- grouping_ Col: Group column name.
- optim_ Params: Optimize parameters.
- num_ all_ Groups: The number of complete groups.
- num_ failed_ Groups: The number of failed training groups.
- total_ rows_ Processed: The number of data rows processed.
- total_ rows_ Skipped: The number of data rows skipped.

5.5.2.5.2.2Predictive function

grammar

The syntax of the prediction function for support vector machines is as follows:

```
svm_ predict(model_table,
                    new_ data_ table,
                    id_ col_ name,
                        output_ table)
```

Parameter Description

- model_ Table: Model result table.
- new_ data_ Table: The new data table to be predicted.
- id_ col_ Name: The ID identification column of the new data table.
- output_ Table: A table that stores the predicted results.

5.5.2.5.2.3Example

- Create a data table for training and insert training data.

```
CREATE TABLE houses (id INT, tax INT, bedroom INT, bath REAL, price INT,
                    size INT, lot INT);
INSERT INTO houses VALUES
( 1,   590,    2,    1,   50000,   770,   22100),
( 2,  1050,    3,    2,   85000,  1410,   12000),
( 3,    20,    3,    1,   22500,  1060,    3500),
```

```
( 4,    870,      2,      2,    90000,   1300,     17500),
( 5,   1320,      3,      2,   133000,   1500,     30000),
( 6,   1350,      2,      1,    90500,    820,     25700),
( 7,   2790,      3,    2.5,   260000,   2130,     25000),
( 8,    680,      2,      1,   142500,   1170,     22000),
( 9,   1840,      3,      2,   160000,   1500,     19000),
(10,   3680,      4,      2,   240000,   2790,     20000),
(11,   1660,      3,      1,    87000,   1030,     17500),
(12,   1620,      3,      2,   118600,   1250,     20000),
(13,   3100,      3,      2,   140000,   1760,     38000),
(14,   2070,      2,      3,   148000,   1550,     14000),
(15,    650,      3,    1.5,    65000,   1450,     12000)
```

- Train a support vector machine classification model.

```
SELECT mllib.svm_ classification(
'test.houses',
'test.houses_ svm',
'price < 100000',
'array double[1, tax, bath, size]',

'max_ iter=20'
);
```

- View training results.

```
gbase> SELECT * FROM houses_ svm\G
*************************** 1. row ***************************
                    coef: 0.103513, -1.17016, -0.0573659, 1.29247
                    loss: 14119.6
  norm_ of_ gradient: 21880
      num_ iterations: 20
num_ rows_ processed: 15
  num_ rows_ skipped: 0
    dep_ var_ mapping: 0,1
1 row in set (Elapsed: 00:00:00.00)

gbase> SELECT * FROM houses_ svm_ summary\G
```

```
*************************** 1. row ***************************
                    method: SVC
              source_ table: test.houses
                model_ table: test.houses_ svm
      dependent_ varname: price < 100000
  independent_ varname: array double[1, tax, bath, size]
              grouping_ col: NULL
              optim_ params: init_ stepsize=0.01,
                             decay_ factor=0.9,
                             max_ iter=20,
                             tolerance=1e-10,
                             epsilon=0.01,
                             eps_ table=,
                             class_ weight=
           num_ all_ groups: 1
       num_ failed_ groups: 0
total_ rows_ processed: 15
    total_ rows_ skipped: 0
1 row in set (Elapsed: 00:00:00.00)
```

- Use a model for prediction.

```
gbase> SELECT mllib.svm_ predict('test.houses_svm', 'test.houses', 'id', 'test.houses_pred')
as result;
+---------+
| result    |
+---------+
| Success |
+---------+
1 row in set (Elapsed: 00:00:00.02)

gbase> SELECT id, prediction, (price < 100000) as pred_ target FROM houses JOIN
houses_ pred USING (id) ORDER BY id;
+------+------------+-------------+
| id      | prediction | pred_ target |
+------+------------+-------------+
|       1 |                  1 |                   1 |
```

```
|      2 |            1 |            1 |
|      3 |            1 |            1 |
|      4 |            1 |            1 |
|      5 |            1 |            0 |
|      6 |            0 |            1 |
|      7 |            0 |            0 |
|      8 |            1 |            0 |
|      9 |            0 |            0 |
|     10 |            0 |            0 |
|     11 |            0 |            1 |
|     12 |            0 |            0 |
|     13 |            0 |            0 |
|     14 |            0 |            0 |
|     15 |            1 |            1 |
+------+------------+-------------+
15 rows in set (Elapsed: 00:00:00.00)
```

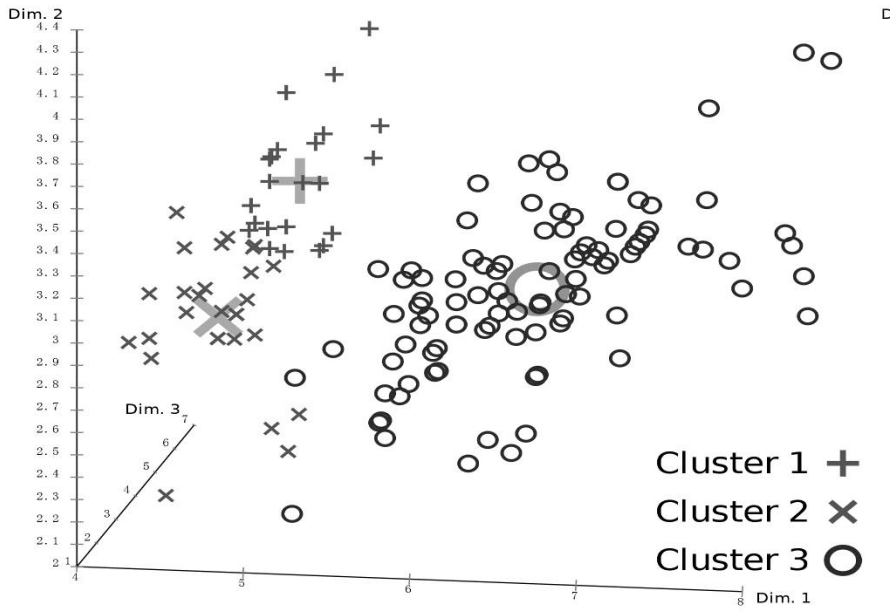
The above prediction query statement uses a support vector machine model to determine whether the house price is less than 100000. The id column represents the house number, the prediction column represents the prediction result (1 represents less than 100000, 0 represents no less than 100000), and the pred column_ The target is the actual situation, and from the output results, it can be seen that the support vector machine model is basically correct.

5.5.2.6 clustering algorithm

5.5.2.6.1 k-means clustering

The k-Means algorithm is a type of clustering algorithm that aims to divide n sample points into k clusters, so that each sample point belongs to the cluster corresponding to its nearest centroid, which serves as the clustering criterion, as shown in the following figure:

Figure -516 k-Means clustering



5.5.2.6.2 **Clustering function**

5.5.2.6.2.1K-Means algorithm

The k-Means algorithm classifies the original points by calculating and comparing the distance between each point and a specified seed point (centroid point), calculates a new centroid point, iterates repeatedly until the group change is less than the expected value or the iteration count is full, and then ends the iteration and generates a result table.

grammar

The syntax of the kmeans function is as follows:

```
kmeans ( source_table,
                    expr_ point,
                    initial_ centroids,
                    fn_ dist,

                       max_ num_ iterations,
                    min_ frac_ reassinged
                       )
```

Parameter Description

- source_ Table: An input table containing training data, in the form of "database name. table name", which cannot be omitted in order to establish a result table in the corresponding database.
- expr_ Point: An expression used to calculate coordinate points from a table.
- initial_ Centroids: A two-dimensional array containing the initial centroid points.
- fn_ Dist: The name of the function used to calculate the distance between points, as shown in. Squared is recommended_ dist_ norm2。
- max_ num_ Iterations: The maximum number of iterations.
- min_ frac_ Threatened: Tolerance, if changes are found in the grouping during the iteration process. If the proportion of points to the total number of points is less than this value, the iteration ends.

Result Table Explanation

After the clustering function is successfully executed, a result table will be created to save the model information. The result table contains the following fields to represent the model information:

- Iteration: The number of iterations performed.
- Centroids: Centroids formed by clustering.
- frac_ Reassigned: The proportion of points where the grouping changes at the end of the iteration to the total number of points.

The k-Means algorithm heavily relies on the initial seed, and different seeds sometimes yield different results. Currently, GBMLLib provides two additional algorithms, kmeans_ Random and kmeanspp algorithms: Random and kmeans++algorithms are used to automatically select initial seeds and start k-Means iteration, respectively. The result tables generated by these two methods are the same as those introduced in this section.

5.5.2.6.2.2kmeans_ random

kmeans_ The random algorithm starts iteration by randomly selecting seed points, which is simple to implement. However, if the initial seed selection is not appropriate, the clustering results may not be ideal.

grammar

kmeans_ The syntax of the random function is as follows:

```
kmeans_ random( source_table,
                expr_ point,

                k,
                fn_ dist,

                    max_ num_ iterations,
                min_ frac_ reassinged
                 )
```

Parameter Description

- source_ Table: An input table containing training data, in the form of "database name. table name", which cannot be omitted in order to establish a result table in the corresponding database.
- expr_ Point: An expression used to calculate coordinate points from a table.
- k: The final number of class clusters.
- fn_ Dist: The name of the function used to calculate the distance between points, as shown in. Squared is recommended_ dist_ norm2。
- max_ num_ Iterations: The maximum number of iterations.
- min_ frac_ Threatened: Tolerance. If it is found that the proportion of the number of points with changes in the group to the total number of points during the iteration process is less than this value, the iteration will end.

5.5.2.6.2.3kmeanspp

KMeanspp uses the k-means++algorithm to select seed points and start iteration. The k-means++algorithm improves classification results by making the initial seed points as discrete as possible.

grammar

The syntax of a function is as follows:

```
kmeanspp( source_table,
          expr_ point,

          k,
```

```
                fn_ dist,

                  max_ num_ iterations,
              min_ frac_ reassinged,
                  seeding_ sample_ rate
                    )
```

Parameter Description

- source_ Table: An input table containing training data, in the form of "database name. table name", which cannot be omitted in order to establish a result table in the corresponding database.
- expr_ Point: An expression used to calculate coordinate points and array types from a table.
- k: The final number of class clusters.
- fn_ Dist: The name of the function used to calculate the distance between points, as shown in. Squared is recommended_ dist_ norm2。
- max_ num_ Iterations: The maximum number of iterations.
- min_ frac_ Threatened: Tolerance. If it is found that the proportion of the number of points with changes in the group to the total number of points during the iteration process is less than this value, the iteration will end.
- seeding_ sample_ Rate: Sampling rate, range of values (0, 1.0). If it is 1.0, all data will be used for sampling; if it is less than 1.0, only partial data will be used.

5.5.2.6.2.4distance function

The k-Means algorithm calculates the distance between points through a distance function. Currently, the built-in functions include:

- squared_ dist_ norm2

  The square of the Euclidean distance is calculated as follows:

  $\|x-y\|_2^2 = \sum_{i=1}^{n}(x_i - y_i)^2$

- dist_ norm2

  The Euclidean distance is calculated using the following formula:

  $\|x-y\|_2 = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$

- dist_ norm1

  The calculation formula for L1 modulus distance is as follows:

  $\|x-y\|_1 = \sum_{i=1}^{n} |x_i - y_i|$

- dist_ angle

  The cosine value of the included angle is calculated using the following formula:

  $\arccos\left(\frac{\langle \vec{x},\vec{y} \rangle}{\|\vec{x}\| \cdot \|\vec{y}\|}\right)$

- dist_ tanimotoo

  The similarity value of tanimoto is calculated using the following formula:

  $1 - \frac{\langle \vec{x},\vec{y} \rangle}{\|\vec{x}\|^2 \cdot \|\vec{y}\|^2 - \langle \vec{x},\vec{y} \rangle}$

5.5.2.6.2.5Example

- Create a data table for training and insert training data.

```
DROP TABLE IF EXISTS km_ sample;
CREATE TABLE km_ sample
    (
        pid INT,
        points BLOB COMMENT 'gbase_ array_ type double[]'
    );

INSERT INTO km_ sample (pid, points)
    VALUES
    (1, ARRAY DOUBLE [1100, 1100]),
    (2, ARRAY DOUBLE [1220, 1110]),
    (3, ARRAY DOUBLE [-1080, 1190]),
    (4, ARRAY DOUBLE [1100, -1100]),
    (5, ARRAY DOUBLE [1080, -1190]),
    (6, ARRAY DOUBLE [1080, 1190]),
    (7, ARRAY DOUBLE [1220, -1110]),
    (8, ARRAY DOUBLE [-1100, -1100]),
    (9, ARRAY DOUBLE [-1080, -1190]),
    (10, ARRAY DOUBLE [-1220, -1110]),
```

```
(11, ARRAY DOUBLE [1300, 1400]),
(12, ARRAY DOUBLE [-1300, -1400]),
(13, ARRAY DOUBLE [-1100, 1100]),
(14, ARRAY DOUBLE [1301, -1400]),
(15, ARRAY DOUBLE [-1220, 1110]),
(16, ARRAY DOUBLE [-1300, 1400])
;
```

- Cluster the input, using kmeanspp as an example:

```
SELECT Mllib.kmeanspp ('madtest.km_sample',
        'points',
        4,
        'squared_ dist_ norm2',

        20, 0.001, 1.0);
```

- View the result table.

```
gbase> select iteration, array_ text(centroids), frac_ reassigned from km_ sample_ result
\G;
*************************** 1. row ***************************
            iteration: 3
array_ text(centroids): {{-1175,1200},{-1175,-1200},{1175.25,-1200},{1175,1200}}
      frac_ reassigned: 0
1 row in set (Elapsed: 00:00:00.00)
```

- Through closest_ The column function displays specific groups:

```
gbase>SELECT
    _ src.pid AS pid,
    array_ text(_src.points) AS point,
    closest_ column
    (
        (
            SELECT
                rel_ result.centroids
            FROM
                km_ sample_ result as rel_ result
```

```
                ),
            _ src.points,
            'squared_ dist_ norm2',
            'squared_ dist_ norm2'
        )
        AS cluster_ id
FROM km_ sample AS _ src    ORDER BY cluster_ id;

+------+---------------+------------+
| pid    | point                | cluster_ id |
+------+---------------+------------+
|       3 | {-1080,1190}    |                 0 |
|     13 | {-1100,1100}    |                 0 |
|     15 | {-1220,1110}    |                 0 |
|     16 | {-1300,1400}    |                 0 |
|       8 | {-1100,-1100} |                1 |
|       9 | {-1080,-1190} |                1 |
|     10 | {-1220,-1110} |                1 |
|     12 | {-1300,-1400} |                1 |
|       4 | {1100,-1100}    |                 2 |
|       5 | {1080,-1190}    |                 2 |
|       7 | {1220,-1110}    |                 2 |
|     14 | {1301,-1400}    |                 2 |
|       1 | {1100,1100}     |                 3 |
|       2 | {1220,1110}     |                 3 |
|       6 | {1080,1190}     |                 3 |
|     11 | {1300,1400}     |                 3 |
+------+---------------+------------+
```

5.5.2.7 **Decision Tree**

Decision tree is a supervised learning algorithm that can be used for classification and regression. It consists of a tree structure, in which nodes represent the detection of attributes, and branches from nodes represent the detection results. Each leaf node is a class label, and the path from the root to the leaf node defines a set of classification or

regression rules.

Decision implementation process: training functions; Prediction function; Decision tree display function.

5.5.2.7.1 Training function

grammar

The syntax of the decision tree training function is as follows:

```
tree_ train(
training_ table_ name,
output_ table_ name,
id_ col_ name,
dependent_ variable,
list_ of_ features,
split_ criterion,
weights,
max_ depth,
min_ split,
min_ bucket,
n_ bins,
pruning_ params
)
```

Parameter Description

- training_ table_ Name: The name of the input table containing training data.
- output_ table_ Name: Save the result table of the training results.
- id_ col_ Name: The column name containing ID information in the training data, and the values in each row should be unique.
- dependent_ Variable: The column name of the dependent variable. Boolean, integer, and text are considered as classification outputs, while double is considered as regression outputs.
- list_ of_ Features: Column names of arguments separated by commas.

- split_ Criterion: Separation standard. For classification trees, they can be 'gini', 'entropy', 'misclass', and default to' gini '; For regression trees, it can only be 'mse'.
- Weights: Column names that identify the weight of each row of input data.
- max_ Depth: The maximum depth of the tree.
- min_ Split: The minimum number of observations for node bifurcation.
- min_ Bucket: The minimum number of observations for leaf nodes.
- n_ Bins: quantile of continuous independent variable.
- pruning_ Params: pruning parameter, formatted as comma separated key value pairs, supports two types: cp (pruning cost), n_ Folds (number of folds for cross validation).

Result Table Explanation

After the training function is successfully executed, a result table will be created to save the model information, which includes the following fields to represent the model information:

- pruning_ Cp: The cost complexity parameter used to prune the training tree.
- Tree: The model obtained after training, in binary format.
- cat_ levels_ in_ Text: Order level of categorical variable
- cat_ n_ Levels: level of each categorical variable
- tree_ Depth: The maximum depth of the tree after training.

Summary Table Description

After the training, a summary table will be generated, which is called the result table name plus "_ Summary ", the fields in the summary table are explained as follows:

- method：‘tree_ train’
- is_ Classification: The classification decision tree is TRUE, and the regression decision tree is FALSE.
- source_ Table: Training data table name
- model_ Table: Training Result Table Name
- id_ col_ Name: The column name in the training table that contains ID information

- dependent_ Varname: dependent variable name
- independent_ Varnames: Argument names
- cat_ Features: list of categorical variable separated by commas
- con_ Features: List of continuous variables, separated by commas
- total_ rows_ Processed: Number of processed rows
- total_ rows_ Skipped: Skipped rows
- dependent_ var_ Levels: The number of levels for categorical dependent variables
- dependent_ var_ Type: dependent variable type
- input_ Cp: Cost complexity parameter for pruning training trees
- independent_ var_ Types: Argument types, separated by commas

5.5.2.7.2 **Predictive function**

grammar

The syntax of the decision tree prediction function is as follows:

```
tree_ predict(
model,
source,
output,
pred_ type
)
```

Parameter Description

- Model: The table name containing the decision tree model should be a tree_ The output table returned by train.
- Source: The table name of the data table that needs to be predicted.
- Output: The name of the predicted result table.
- pred_ Type: optional, default to 'response'. For regression trees, the output is always the predicted value of the dependent variable; For the classification tree, 'response' will output the category prediction of the dependent variable, and 'prob' will output the probability of each category.

5.5.2.7.3 Decision Tree Display Function

Display a graphical representation of the function output decision tree. The output can be in 'dot' format or text format.

grammar

The syntax of the decision tree display function is as follows:

```
tree_ display(
model_ table,
dot_ format
)
```

Parameter Description

- model_ Table: The table name containing the decision tree model should be a tree_ The output table returned by train.
- dot_ Format: If true, output 'dot' format; if false, output text format.

5.5.2.7.4 Example

- Establish a source data table and insert data to construct a classification problem data source.

```
DROP table IF EXISTS dt_ golf;
CREATE TABLE dt_ golf(
    id integer NOT NULL,
    OUTLOOK varchar(30),
    temperature double,
    humidity double,
    windy varchar(30),
    class varchar(30)
);

INSERT INTO dt_ golf VALUES
(1 ,'sunny',85,85,'false','Dont Play'),
(2 ,'sunny',80,90,'true','Dont Play'),
(3 ,'overcast',83,78,'false','Play'),
```

```
(4 ,'rain',70,96,'false','Play'),
(5 ,'rain',68,80,'false','Play'),
(6 ,'rain',65,70,'true','Dont Play'),
(7 ,'overcast',64,65,'true','Play'),
(8 ,'sunny',72,95,'false','Dont Play'),
(9 ,'sunny',69,70,'false','Play'),
(10,'rain',75,80,'false','Play'),
(11,'sunny',75,70,'true','Play'),
(12,'overcast',72,90,'true','Play'),
(13,'overcast',81,75,'false','Play'),
(14,'rain',71,80,'true','Dont Play'),
(15,'rain',80,83,'false','Play')
;
```

- Call tree_ Train generates a classification decision tree.

```
SELECT mllib.tree_ train('madtest.dt_golf', 'madtest.train_output', 'id', 'class',
'OUTLOOK, temperature, humidity
, windy', 'gini', '', 5, 3, 1, 10,'cp=0');
```

- Call tree_ The display function displays the decision tree structure.

```
SELECT mllib.tree_ display('madtest.train_output', FALSE)
(0)[ 5 10]    OUTLOOK in {overcast}
    (1)[0 4]    * --> Play
    (2)[5 6]    windy in {false}
        (5)[2 5]    OUTLOOK in {overcast,rain}
            (11)[0 4]    * --> Play
            (12)[2 1]    temperature <= 69
                (25)[0 1]    * --> Play
                (26)[2 0]    * --> Dont Play
        (6)[3 1]    * --> Dont Play
```

- Call tree_ The predict function predicts the classification results.

```
SELECT mllib.tree_ predict('madtest.train_output','madtest.dt_golf', 'madtest.prediction_results',
'response')
+-----------------------------------------------------------------------------------------------------+
| mllib.tree_ predict('madtest.train_output', 'madtest.dt_golf', 'madtest.prediction_results',
```

```
'response') |
+-----------------------------------------------------------------------------------------------------+
| Success
|
+-----------------------------------------------------------------------------------------------------+
1 row in set
SELECT * from (SELECT dt_golf.*, prediction_results.predict_class FROM dt_golf JOIN
prediction_results USING(id))    t    order by id
--------------


+----+----------+-------------+----------+-------+-----------+---------------+
| id | OUTLOOK    | temperature | humidity | windy | class          | predict_ class |
+----+----------+-------------+----------+-------+-----------+---------------+
|    1 | sunny       |                 85 |            85 | false | Dont Play | Dont Play          |
|    2 | sunny       |                 80 |            90 | true    | Dont Play | Dont Play          |
|    3 | overcast |                83 |           78 | false | Play           | Play                 |
|    4 | rain          |                 70 |           96 | false | Play           | Play                 |
|    5 | rain          |                 68 |           80 | false | Play           | Play                 |
|    6 | rain          |                 65 |           70 | true    | Dont Play | Dont Play          |
|    7 | overcast |                64 |           65 | true    | Play           | Play                 |
|    8 | sunny       |                 72 |           95 | false | Dont Play | Dont Play          |
|    9 | sunny       |                 69 |           70 | false | Play           | Play                  |
| 10 | rain          |                 75 |           80 | false | Play           | Play                 |
| 11 | sunny       |                 75 |           70 | true    | Play           | Dont Play          |
| 12 | overcast |                72 |           90 | true    | Play           | Play                 |
| 13 | overcast |                81 |           75 | false | Play           | Play                |
| 14 | rain          |                 71 |           80 | true    | Dont Play | Dont Play          |
| 15 | rain          |                 80 |           83 | false | Play           | Play                 |
+----+----------+-------------+----------+-------+-----------+---------------+
```

5.6 EVENT event

An event is a procedural database object that is called at the corresponding time. An event can be called once or started periodically, and it is managed by a specific thread, also known as the "event scheduler".

Events and triggers are similar in that they are triggered when something happens. When

a statement is launched on the database, the trigger is triggered, and the event is initiated based on the scheduled event. Due to their similarity, events are also known as temporary triggers.

Each create event statement creates an event. Each event consists of two main parts. The first part is the event schedule, which represents when and how frequently the event starts; The second part is the event action, which is the code executed when the event starts. The action of the event contains an SQL statement, which may be a simple Insert or UPDATE statement, a stored procedure, or a BEGIN END statement block, both of which allow for the execution of multiple SQL statements.

An event can be active (open) or stopped (closed). Active means that the event scheduler checks whether the event action must be called, and stopped means that the declaration of the event is stored in the directory, but the scheduler does not check whether it should be called. After an event is created, it immediately becomes active, and an active event can be executed once or multiple times.

 **be careful**

- Event currently does not support failover.

5.6.1 event scheduler

Event scheduler event_ The scheduler is responsible for calling events, which are turned on by default. This scheduler constantly monitors whether an event needs to be called. To create an event, the scheduler must be opened.

```
gbase> SHOW VARIABLES LIKE '%event_ scheduler%';
+-----------------+-------+
| Variable_ name      | Value |
+-----------------+-------+
| event_ scheduler | ON        |
+-----------------+-------+
1 row in set (Elapsed: 00:00:00.00)
```

5.6.1.1 Enable Event Scheduler

Opening method

Method 1: You can start the event scheduler through the following command line.

```
SET GLOBAL event_ scheduler = ON;
SET @@global.event_ scheduler = ON;
SET GLOBAL event_ scheduler = 1;
SET @@global.event_ scheduler = 1;
```

Method 2: Use the configuration file gbase_ 8a_ Gcluster.cnf opens the event scheduler.

```
……
[gbased]
……
event_ Scheduler=1 # or ON
……
```

View scheduler threads

```
gbase> show processlist\G
*************************** 1. row ***************************
      Id: 1
    User: event_ scheduler
    Host: localhost
      vc: NULL
      db: NULL
Command: Daemon
    Time: 1034068
   State: Waiting for event lock
    Info: NULL
*************************** 2. row ***************************
      Id: 621
    User: root
    Host: localhost
      vc: vc1
      db: NULL
Command: Sleep
    Time: 376038
   State:
    Info: NULL
2 rows in set (Elapsed: 00:00:00.00)
```

5.6.1.2 Close Event Scheduler

Closing method

Method 1: The event scheduler can be closed by using the following command line.

```
SET GLOBAL event_ scheduler = OFF;
SET @@global.event_ scheduler = OFF;
SET GLOBAL event_ scheduler = 0;
SET @@global.event_ scheduler = 0;
```

Method 2: Use the configuration file gbase_ 8a_ Close the event scheduler with gbase.cnf.

```
……
[gbased]
……
```

```
event_ Scheduler=0 # or OFF, DISABLED
……
```

View scheduler threads

```
gbase> show processlist\G
*************************** 1. row ***************************
        Id: 621
      User: root
      Host: localhost
        vc: vc1
        db: NULL
Command: Sleep
      Time: 376262
     State:
      Info: NULL
*************************** 2. row ***************************
        Id: 883
      User: root
      Host: 172.168.83.11:44008
        vc: NULL
        db: NULL
Command: Sleep
      Time: 6
     State:
      Info: NULL
2 rows in set (Elapsed: 00:00:00.00)
```

5.6.2 Create Event

Grammar format

```
CREATE [DEFINER = { user | CURRENT_USER }] EVENT
[IF NOT EXISTS] <event_ name>
ON SCHEDULE <schedule>
[ON COMPLETION [NOT] PRESERVE]
[ENABLE | DISABLE]
[GLOBAL | LOCAL]
[COMMENT 'comment']
DO event_ body;

schedule:
AT timestamp [+ INTERVAL interval] ... | EVERY interval [STARTS timestamp
[+ INTERVAL interval] ...] [ENDS timestamp [+ INTERVAL interval] ...]
```

```
interval:
quantity { YEAR | QUARTER | MONTH | DAY | HOUR | MINUTE |WEEK |
SECOND | YEAR_MONTH | DAY_HOUR | DAY_MINUTE | DAY_SECOND |
HOUR_MINUTE | HOUR_SECOND | MINUTE_SECOND }
```

Table -5204 Parameter Description

Field Name	Explanation of Meaning
event_ name	Created event name (uniquely determined)
ON SCHEDULE	Plan tasks.
schedule	There are two forms of determining the execution time and frequency of an event (note that the time must be in the future, as past times can cause errors): AT and EVERY.
[ON COMPLETION [NOT] PRESERVE]	Optional, default is ON Completion Not PRESERVE, which automatically drops the event after the scheduled task is executed; ON Completion PRESERVE will not drop.
COMMENT 'comment'	Optional, comment is used to describe an event; Equivalent to a comment, with a maximum length of 64 bytes.
[ENABLE | DISABLE]	Set the status of the event. By default, ENABLE indicates that the system is attempting to execute the event, and DISABLE indicates that the event is closed and can be modified using alter.
[GLOBAL | LOCAL]	GLOBAL represents exclusive execution by each node, while LOCAL represents independent execution by each node
DO event_ body	SQL statements that need to be executed (can be compound statements). The use of CREATE EVENT in stored procedures is legal.

be careful

- The default creation event is stored in the current library, and it can also display which library the specified event is created in;
- Only the events created in the current library can be viewed through show events;
- After the event is executed, it is released. If the event is executed immediately, it will be automatically deleted. Multiple calls to the event or waiting for execution can be viewed;
- If two events need to be called at the same time, GBase will determine the order in which they are called. If the order is to be specified, it is necessary to ensure that one event is executed at least 1 second after the other event;
- For recursively scheduled events, the end date cannot be before the start date;
- Select can be included in an event, but its result disappears as if it has not been executed;
- The load statement is not supported when creating events;
- Event does not support the creation of procedures, functions, triggers, and other objects.

Example

Create a test table:

```
gbase> DROP TABLE IF EXISTS events_ list;
gbase> CREATE TABLE events_ list(event_name VARCHAR(20) NOT NULL,
event_ started TIMESTAMP NOT NULL);
```

Create event 1 (start event immediately):

```
CREATE EVENT event_ now
ON SCHEDULE
AT NOW()
DO insert into events_ list values('event_now', now());
```

To view event execution results:

```
gbase> select * from events_ list;
+------------+---------------------+
| event_ name | event_ started      |
+------------+---------------------+
| event_ now  | 2017-07-01 04:06:40 |
+------------+---------------------+
```

Create event 2 (start event every minute):

```
CREATE EVENT test.event_ minute
ON SCHEDULE
EVERY  1 MINUTE
```

```
DO insert into events_ list values('event_now', now());
To view event execution results:
gbase> SELECT * FROM events_ list;
+-------------+---------------------+
| event_ name    | event_ started         |
+-------------+---------------------+
| event_ now    | 2017-07-01 04:26:53 |
| event_ now    | 2017-07-01 04:27:53 |
| event_ now    | 2017-07-01 04:28:53 |
+-------------+---------------------+
```

5.6.3 Viewing Events

Grammar format

To view events in the current library:

```
SHOW EVENTS;
```

View all events:

```
SELECT * FROM gbase.event;
SELECT * FROM information_ schema.events;
```

5.6.4 Modify Event

Function Description

An alter event statement can modify the definition and properties of an event. We can make an event stop or make it active again, or modify the name of an event or the entire schedule. However, when an event defined using the ON Completion NOT PRESERVE attribute is executed for the last time, the event no longer exists and cannot be modified.

Grammar format

```
ALTER
    EVENT event_ name
    [ON SCHEDULE schedule]
    [ON COMPLETION [NOT] PRESERVE]
    [RENAME TO new_event_name]
    [ENABLE | DISABLE]
    [COMMENT 'comment']
    [DO event_body]
```

Example

```
Modify the start event per minute in the above example to start every 30 seconds:
ALTER EVENT test.event_ minute
  ON SCHEDULE
EVERY   30 SECOND
DO INSERT INTO events_ list VALUES('event_now', now());
Change the event name of the above example to event_ second:
ALTER EVENT test.event_ minute
RENAME TO test.event_ second;
Modify the events in the above example to be inactive and inactive again:
ALTER EVENT test.event_ second DISABLE;
ALTER EVENT test.event_ second ENABLE;
```

5.6.5 Delete Event

Function Description

If an event is no longer needed, we can use a drop event statement to delete it. We don't need to wait until the last event call to use this statement.

Grammar format

```
DROP EVENT [IF EXISTS] event_ name;
```

Example

```
drop event if exists event_ second;
```

5.7 System Table

This document describes the system tables in GBase 8a MPP Cluster. System tables exist in four databases, namely information_ Schema, gbase, gclusterdb, and performance_ schema。

- information_ The information in the schema is metadata information, which is obtained through organizing related objects and not pre stored information;
- The gbase inventory stores some persistence information, which is stored in the table of the GsSYS engine;
- Gclusterdb stores information that needs to be saved across the cluster, using the express engine;
- performance_ The schema stores running status information.

5.7.1 information_ Schema library

5.7.1.1 Table Function Definition

Table 5205 Table Function Information Description:

Table Name	Description
CHARACTER_ SETS	The character set table provides information on the character sets that can be used in the instance. The show character set result is taken from this table
COLLATIONS	Provides comparative information about each character set
COLLATION_ CHARACTER_ SET_ APPLICABILITY	Indicates the character set that can be used for proofreading, equivalent to the first two fields of show collation
COLUMNS	The column information in the table is provided, and the result of showing columns from schemaname. tablename is taken here
COLUMN_ PRIVILEGES	Column permission table, providing information about column permissions.
ENGINES	Record information about the engine
EVENTS	A time trigger that triggers related SQL statements or stored procedures at a specific point in time, different from an event trigger
FILES	File related information used to store tablespace data
FUNCTIONS	Display udf function information within the system
GLOBAL_ STATUS	Record the global status information of the system
GLOBAL_	Record system global variable setting information

VARIABLES	
KEY_ COLUMN_ USAGE	Record the constraint information of key columns in the database.
PARTITIONS	Record table partition information in the database.
PROCESSLIST	The information about a client connection, where the show processlist command is taken
PROFILING	The resource consumption information for executing SQL statements requires setting the profiling parameter to 1
REFERENTIAL_ CONSTRAINTS	Record foreign key information, currently empty.
ROUTINES	Provides information about storing subroutines (stored programs and functions). Routines table does not contain custom functions (UDFs)
COORDINATORS_ TASK_ INFORMATION	Record the execution information of SQL statements at the management node level.
GNODES_ TASK_ INFORMATION	Record the resource usage information of SQL statements on data nodes.
COORDINATORS_ RESOURCE_ POOL_ USAGE	Record the real-time usage of resource pools for all coordinator nodes, including the number of waiting tasks, running tasks, etc
GNODES_ RESOURCE_ POOL_ USAGE	Record the real-time usage of resource pools for data nodes, including CPU, memory, and disk usage information
GNODES_ USER_ DISKSPACE_ USAGE	Record the usage of user disk quotas, including user limited disk space, actual usage of disk space, etc
COORDINATORS_ RESOURCE_ POOL_ STATUS	Record the historical usage of resource pools in the cluster layer, including the number of running tasks, average task running time, average task waiting time, etc
GNODES_ RESOURCE_ POOL_ STATUS	Record the historical usage of resource pools for data nodes, including CPU, memory, and disk usage information
RESOURCE_ POOL_ EVENTS	Record cluster layer resource pool event information
COORDINATOR_ RESOURCE_ POOL_ USAGE	Record the real-time usage of the resource pool of the current coordinator node, including the number of waiting tasks, running tasks, etc
SCHEMATA	Provide information about all databases in the current instance, and obtain this table by showing the results of databases

SCHEMA_ PRIVILEGES	The database permission table provides information about database permissions.
SESSION_ STATUS	Current session status information
SESSION_ VARIABLES	The variable name and value of the current session
STATISTICS	Provides information about table indexes, where the result of show index from schemaname. tablename is taken
TABLES	Provide information about the tables in the database (including views), detailing which library, table type, table engine, creation time, etc. a table belongs to. The results of showing tables from schema are obtained here
TABLE_ CONSTRAINTS	Describing the tables with constraints and their constraint types
TABLE_ PRIVILEGES	Table Permission Table
TRIGGERS	Provides information about triggering programs, and must have super permission to view this table
USER_ PRIVILEGES	The user permission table provides information about global permissions
LOAD_ STATUS	Record loading progress information
LOAD_ TRACE	Record the loading error log information of the current coordinator node
CLUSTER_ LOAD_ TRACE	Record the loading error log information of all coordinator nodes
LOAD_ RESULT	Record the loading result information of the current coordinator node
CLUSTER_ LOAD_ RESULT	Record the loading result information of all coordinator nodes
VIEWS	Providing information about views in the database requires show views permission, otherwise view information cannot be viewed
VC	Record virtual cluster information
CACHES	Record cache information
PRIORITIES	Priority status display (related to cgroups and priority queues)
TABLE_ LOCKS	Used to display the usage of the current table lock

MEMORY_ HEAP_ INFO	Information on each heap in memory
CACHE_ ACCESS_ INFO	The memory access status of the select/insert/update operation, the delete/show operation table value does not change, and the gbased restart table value returns to zero
CLUSTER_ TABLES	Record the disk space information occupied by all shards
CLUSTER_ TABLE_ SEGMENTS	Record the disk space occupied by each shard's data
KAFKA_ CONSUMER_ STATUS	Record the running status of transactional consumer tasks
KAFKA_ CONSUMER_ EFFICIENCY	Record the operational efficiency of transactional consumer tasks, including time consumption statistics for each stage
KAFKA_ LOADER_ CONSUMER_ STATUS	Record the running status of the loaded consumer task
SYS_ SCN	System change number
DML_ INFO	Record information on the number of running dml statements
ALL_ ENCRYPTION_ CERTIFICATE_ STATUS	Record the current encryption status information of all nodes
COORDINATOR_ ENCRYPTION_ CERTIFICATE_ STATUS	Record the current encryption status information of the cluster layer
GNODE_ ENCRYPTION_ CERTIFICATE_ STATUS	Record the current encryption status information of all node points
ENCRYPTION_ CERTIFICATE_ STATUS	Record the current encryption status information of this node
TABLESPACES	Record tablespace related information
TABLESPACE_ NAMES	Record the name, path, and other related information of the tablespace.
CLUSTER_ MONIT_ INFO	Collect cluster status information.

5.7.1.2 Table Structure Description

5.7.1.2.1 **CHARACTER_ SETS**

Function Description

This table provides information on the character sets that can be used in the instance, and the show character set results are taken from this table.

Table Structure Description

Table 5206 Table Structure Information Explanation:

Column Name	Description
CHARACTER_ SET_ NAME	Character Set Name
DEFAULT_ COLLATE_ NAME	Default Proofreading Name
DESCRIPTION	Character Set Category Description
MAXLEN	Maximum length of character set

5.7.1.2.2 COLLATIONS

Function Description

This table provides comparative information about each character set.

Table Structure Description

Table 5207 Table Structure Information Explanation:

Column Name	Description
COLLATION_ NAME	The name of the proofreading set
CHARACTER_ SET_ NAME	Character sets related to proofreading sets
ID	Proofreading set ID
IS_ DEFAULT	Is the proofreading set the default proofreading set for the character set
IS_ COMPILED	Is the character set compiled into the server
SORTLEN	Related to the memory size required for strings represented by character sets

5.7.1.2.3 COLLATION_ CHARACTER_ SET_ APPLICABILITY

Function Description

Indicates the character set that can be used for proofreading, equivalent to the first two fields of show collation.

Table Structure Description

Table 5208 Table Structure Information Explanation:

Column Name	Description
COLLATION_ NAME	Proofread Set Name
CHARACTER_ SET_ NAME	Character sets related to proofreading sets

5.7.1.2.4 COLUMNS

Function Description

The column information in the table is provided, and the result of showing columns from schemaname. tablename is taken here.

Table Structure Description

Table -5209 Table Structure Information Explanation:

Column Name	Description
TABLE_ CATALOG	Table of Contents for Registration
TABLE_ VC	Name of the virtual cluster to which it belongs
TABLE_ SCHEMA	Database name
TABLE_ NAME	Table Name
COLUMN_ NAME	Column Name
ORDINAL_ POSITION	Which column is the field in the table
COLUMN_ DEFAULT	Default data for columns
IS_ NULLABLE	Can the field be empty
DATA_ TYPE	data type
CHARACTER_ MAXIMUM_ LENGTH	Maximum length of characters
CHARACTER_ OCTET_ LENGTH	Byte length of characters
NUMERIC_ PRECISION	Data accuracy
NUMERIC_ SCALE	Data scale
CHARACTER_ SET_ NAME	Character Set Name
COLLATION_ NAME	Character set validation name
COLUMN_ TYPE	Column Type
COLUMN_ KEY	Key column [NULL | MUL | PRI]
EXTRA	Additional description [NULL | on update Current_TIMESTAMP | auto_increase]
PRIVILEGES	Field operation permissions [select | select, insert, update, references]
COLUMN_ COMMENT	Field comments and descriptions

5.7.1.2.5 COLUMN_ PRIVILEGES

Function Description

Column permission table, providing information about column permissions.

Table Structure Description

Table 5210 Table Structure Information Explanation:

Column Name	Description
GRANTEE	User name granted permission
TABLE_ CATALOG	Table of Contents for Registration
TABLE_ VC	The virtual cluster name of the database to which the column belongs to the table
TABLE_ SCHEMA	The database name to which the column belongs to the table
TABLE_ NAME	Table name where the column is located
COLUMN_ NAME	Column Name
PRIVILEGE_ TYPE	Permission Type
IS_ GRANTABLE	Do you have the authority to grant permissions

5.7.1.2.6 ENGINES

Function Description

Information about the engine.

Table Structure Description

Table 5211 Table Structure Information Explanation:

Column Name	Description
ENGINE	Engine Name
SUPPORT	Does the database support the use of
COMMENT	A brief introduction to the engine
TRANSACTIONS	Does it support transaction processing [YES | NO]
XA	Support Distributed transaction Processing [YES | NO]
SAVEPOINTS	Does it support the savepoint rollback method for transactions

5.7.1.2.7 EVENTS

Function Description

A time trigger that triggers relevant SQL statements or stored procedures at a specific point in time.

Table Structure Description

Table 5212 Table Structure Information Explanation:

Column Name	Description
EVENT_ CATALOG	Time Trigger Registration Directory Name
EVENT_ VC	The virtual cluster name where the time trigger is located
EVENT_ SCHEMA	The database where the time trigger is located
EVENT_ NAME	The name of the time trigger
DEFINER	Creator of Time Trigger
TIME_ ZONE	time zone
EVENT_ BODY	Time Trigger Principal Category [SQL]
EVENT_ DEFINITION	Time trigger definition, i.e. defining code
EVENT_ TYPE	Type, [RECURRING |...] represents repeated execution, [ONE TIME] represents execution once
EXECUTE_ AT	Is NULL
INTERVAL_ VALUE	Execution time interval
INTERVAL_ FIELD	Unit of time interval
SQL_ MODE	Execution mode of SQL
STARTS	Start execution time
ENDS	Execution end time
STATUS	Is the time trigger available, [ENABLED | DISABLED]
ON_ COMPLETION	After execution is completed, do you want to keep it? If not, drop it. [PRESERVE | NOT PRESERVE]
CREATED	Creation time
LAST_ ALTERED	Last modification time
LAST_ EXECUTED	Last execution time
EVENT_ COMMENT	remarks
EXECUTE_ MODE	The execution mode of Event, where 0 indicates that the event is executed in isolate mode and 1 indicates that it is executed in exclusive mode, with a default value of 0.
CHARACTER_ SET_ CLIENT	Character Set Encoding
COLLATION_ CONNECTION	Linking Character Set Comparison Method
DATABASE_ COLLATION	Character Set Comparison Method for Database

5.7.1.2.8 FILES

Function Description

File related information used to store tablespace data.

Table Structure Description

Table 5213 Table Structure Information Explanation:

Column Name	Description
FILE_ ID	The ID of the tablespace
FILE_ NAME	The name of the data file
FILE_ TYPE	The file type of the tablespace, [TABLESPACE | TEMPORARY | UNDO LOG]
TABLESPACE_ NAME	SQL name of the tablespace
TABLE_ CATALOG	Register directory
TABLE_ SCHEMA	Database name
TABLE_ NAME	Table Name
LOGFILE_ GROUP_ NAME	Log or data belongs to a log file group
LOGFILE_ GROUP_ NUMBER	The automatic generation ID of the log file group to which the log or data belongs
ENGINE	Storage Engine
FULLTEXT_ KEYS	Always NULL
DELETED_ ROWS	Always NULL
UPDATE_ COUNT	Always NULL
FREE_ EXTENTS	The number of empty extensible regions in the current data file
TOTAL_ EXTENTS	The number of expandable areas that have been used in the current data file
EXTENT_ SIZE	Expand partition size
INITIAL_ SIZE	The initial size of the file, in bytes
MAXIMUM_ SIZE	Maximum file size
AUTOEXTEND_ SIZE	Automatically expanded size
CREATION_ TIME	Creation time
LAST_ UPDATE_ TIME	Last update time
LAST_ ACCESS_ TIME	Last visit time
RECOVER_ TIME	recovery time
TRANSACTION_ COUNTER	Always NULL

VERSION	edition
ROW_ FORMAT	Row format
TABLE_ ROWS	How many rows of data are there in the table
AVG_ ROW_ LENGTH	Average row length
DATA_ LENGTH	Data length
MAX_ DATA_ LENGTH	Maximum data length
INDEX_ LENGTH	Index length
DATA_ FREE	All available space
CREATE_ TIME	Creation time
UPDATE_ TIME	Update time
CHECK_ TIME	Inspection time
CHECKSUM	Checksum
STATUS	Default NORMAL [NORMAL | IMPORTING]
EXTRA	Other, varchar (255)

5.7.1.2.9 FUNCTIONS

Function Description

Display udf function information within the system.

Table Structure Description

Table -5214 Table Structure Information Explanation:

Column Name	Description
FUNC_ NAME	UDF Function Name
RETURN_ TYPE	Function return name
FUNC_ TYPE	Function type, UDF

5.7.1.2.10 GLOBAL_ STATUS

Function Description

Record the global status information of the system.

Table Structure Description

Table -5215 Table Structure Information Explanation:

Column Name	Description

VARIABLE_ NAME	Variable Name
VARIABLE_ VALUE	Variable value
SESSION_ LEVEL	Is it a session level state
WRITEABLE	Can changes be made during service operation

5.7.1.2.11 GLOBAL_ VARIABLES

Function Description

Record system global variable setting information.

Table Structure Description

Table 5216 Table Structure Information Explanation:

Column Name	Description
VARIABLE_ NAME	Variable Name
VARIABLE_ VALUE	Variable value
SESSION_ LEVEL	Is it a session level state
WRITEABLE	Can changes be made during service operation

5.7.1.2.12 KEY_ COLUMN_ USAGE

Function Description

Record the constraint information of key columns in the database.

Table Structure Description

Table 5217 Table Structure Information Explanation:

Column Name	Description
CONSTRAINT_ CATALOG	The directory name to which the constraint belongs, always NULL
CONSTRAINT_ SCHEMA	Constraint belongs to database name
CONSTRAINT_ NAME	Constraint Name
TABLE_ CATALOG	The directory name of the table containing constraints, always NULL
TABLE_ VC	The name of the virtual cluster to which the constraint belongs in the table
TABLE_ SCHEMA	The database name of the table where the constraint belongs
TABLE_ NAME	Constraint table name
COLUMN_ NAME	Constraint Column Name
ORDINAL_ POSITION	Position listed within constraints, written

	starting from 1
POSITION_ IN_ UNIQUE_ CONSTRAINT	NULL represents a unique primary key constraint.
REFERENCED_ TABLE_ SCHEMA	Constraint Referenced Data Name
REFERENCED_ TABLE_ NAME	Table name referenced by constraint
REFERENCED_ COLUMN_ NAME	List of constraint references

5.7.1.2.13 PARTITIONS

Function Description

Record table partition information in the database.

Table Structure Description

Table -5218 Table Structure Information Explanation:

Column Name	Description
TABLE_ CATALOG	The directory where the table is located is always NULL
TABLE_ SCHEMA	Database name where the table is located
TABLE_ NAME	Table Name
PARTITION_ NAME	Partition name
SUBPARTITION_ NAME	If the partition table row represents a sub partition, it is the sub partition name; otherwise, it is 0; otherwise, it is NULL
PARTITION_ ORDINAL_ POSITION	The position number of the partition index, starting from 1. The index order of all partitions is the same as the defined order
SUBPARTITION_ ORDINAL_ POSITION	Location number of sub partitions
PARTITION_ METHOD	Partition type, with values including LIST, RANGE, HASH, KEY, etc
SUBPARTITION_ METHOD	The type of word partition, with values including LIST, RANGE, HASH, KEY, etc
PARTITION_ EXPRESSION	Create the current partition scheme for the table
SUBPARTITION_ EXPRESSION	Create the current sub partition scheme for the table, if it is not NULL
PARTITION_ DESCRIPTION	Used for RANGE and LIST partitions. For RANGE partitions, it contains the value set in the partition VALUES LESS THAN clause, which can be an integer

	or MAXVALUE. For a LIST partition, this column contains the values defined in the VALUES IN clause of the partition, which is a comma separated list of integer values.
TABLE_ ROWS	Number of rows in the partition
AVG_ ROW_ LENGTH	The average length of rows in a partition or sub partition, in bytes
DATA_ LENGTH	The total length of all rows stored in a partition or sub partition, in bytes
MAX_ DATA_ LENGTH	The maximum number of bytes stored in a partition or sub partition
INDEX_ LENGTH	The length of the index file for a partition or sub partition, in bytes
DATA_ FREE	Number of unused bytes allocated to a partition or sub partition
CREATE_ TIME	Creation time of partition or sub partition
UPDATE_ TIME	Update time of partition or sub partition
CHECK_ TIME	The last time the table to which the partition or sub partition belongs was checked
CHECKSUM	Checksum value, no non NULL
PARTITION_ COMMENT	Partition annotation, if none, display as empty
NODEGROUP	Node group to which the partition belongs
TABLESPACE_ NAME	Name of the tablespace to which the partition belongs
TABLE_ ID	Partition's Table ID

5.7.1.2.14 PROCESSLIST

Function Description

The show processlist command retrieves information about a client connection.

Table Structure Description

Table -5219 Table Structure Information Explanation:

Column Name	Description
ID	ID number of the processlist
TASKID	Task ID
SUBTASKID	Subtask ID
THREADID	Thread ID
USER	Connected Users
HOST	IP number: Port number
VC	The virtual cluster being used by the process

DB	The database being used by the process
COMMAND	The command executed by the current connection [Query | Connect | Sleep]
START_ TIME	Task start time
TIME	The duration of the current state
STATE	Display the execution status of SQL statements using the current connection
RESOURCE_ POOL_ NAME	Name of the resource pool to which the task belongs
RESOURCE_ POOL_ ID	The resource pool ID to which the task belongs, with the same value range as the field type value range
RESOURCE_ POOL_ PRIORITY	Priority of the resource pool to which the task belongs, value range [1, 8]
WAITING_ TIME	The waiting time for the current task, with the same value range as the field type, in seconds
RUNNING_ TIME	The current task runtime, with the same value range as the field type, in seconds
LOCK	Is there a lock
WAIT	Waiting for events
INFO	Show executing SQL statements
TRACE	Trace information

5.7.1.2.15 PROFILING

Function Description

The resource consumption information for executing SQL statements requires setting the profiling parameter to 1.

Table Structure Description

Table -5220 Table Structure Information Explanation:

Column Name	Description
QUERY_ ID	The sequence number of executing SQL
SEQ	Same QUERY_ Number of sub processes with different IDs
STATE	The sub states of each SQL execution
DURATION	Execution time of sub states
CPU_ USER	User CPU usage time
CPU_ SYSTEM	System CPU usage time
CONTEXT_ VOLUNTARY	Active environment switching
CONTEXT_ INVOLUNTARY	Passive environment switching
BLOCK_ OPS_ IN	Number of input blocks

BLOCK_ OPS_ OUT	Number of output blocks
MESSAGES_ SENT	Number of interactive messages sent
MESSAGES_ RECEIVED	Number of interactive information received
PAGE_ FAULTS_ MAJOR	Number of major page errors
PAGE_ FAULTS_ MINOR	Number of secondary page errors
SWAPS	How many exchanges have occurred

5.7.1.2.16 REFERENTIAL_ CONSTRAINTS;

Function Description

Record foreign key information, currently empty.

Table Structure Description

Table 5-221 Table Structure Information Explanation:

Column Name	Description
CONSTRAINT_ CATALOG	Constraint Catalog
CONSTRAINT_ SCHEMA	Constraint database
CONSTRAINT_ NAME	Constraint Name
UNIQUE_ CONSTRAINT_ CATALOG	Directory containing unique constraints referenced by constraints
UNIQUE_ CONSTRAINT_ SCHEMA	Database containing unique constraints referenced by constraints
UNIQUE_ CONSTRAINT_ NAME	The name of the unique constraint applied by the constraint
MATCH_ OPTION	Constrain the value of the match attribute
UPDATE_ RULE	Constrain the value of the ON UPDATE attribute
DELETE_ RULE	Constrain the value of the ON DELETE attribute
TABLE_ NAME	Constraint Table Name
REFERENCED_ TABLE_ NAME	Table name referenced by constraint

5.7.1.2.17 ROUTINES

Function Description

Provides information about storing subroutines (stored programs and functions), and the routing table does not contain custom functions (UDFs).

Table Structure Description

Table 5-222 Table Structure Information Explanation:

Column Name	Description
SPECIFIC_ NAME	Function or stored procedure name
ROUTINE_ CATALOG	Registration table name, NULL
ROUTINE_ VC	Virtual cluster name where the function or stored procedure is located
ROUTINE_ SCHEMA	The library name where the function or stored procedure is located
ROUTINE_ NAME	Function or stored procedure name
ROUTINE_ TYPE	Type [PROCESS | Function]
DTD_ IDENTIFIER	Data type description
ROUTINE_ BODY	SQL
ROUTINE_ DEFINITION	The content of a function or stored procedure
EXTERNAL_ NAME	NULL
EXTERNAL_ LANGUAGE	NULL
PARAMETER_ STYLE	[SQL]
IS_ DETERMINISTIC	Is it deterministic
SQL_ DATA_ ACCESS	[CONTAINS SQL| ...]
SQL_ PATH	NULL
SECURITY_ TYPE	Security type [DEFINER |...]
CREATED	Creation time
LAST_ ALTERED	Last modified time
SQL_ MODE	Execution mode of SQL
ROUTINE_ COMMENT	Remarks
DEFINER	creator
CHARACTER_ SET_ CLIENT	The character set used by the session during creation
COLLATION_ CONNECTION	The validation set used by the session during creation
DATABASE_ COLLATION	Verification set of related databases

5.7.1.2.18 COORDINATORS_ TASK_ INFORMATION

Function Description

Record the execution information of SQL statements at the management node level.

Table Structure Description

Table -5223 Table Structure Information Explanation:

Column Name	Description
COORDINATOR_ NAME	Management node name
VC	Name of the virtual cluster to which it belongs
ID	SessionID

TASKID	Task ID
SUBTASKID	Subtask ID
THREADID	Thread ID
USER	Execute User
HOST	Execution node
DB	Affiliated database
COMMAND	SQL Type
START_ TIME	Start execution time
TIME	execution time
STATE	SQL Execution Status
RESOURCE_ POOL_ NAME	Dynamic Resource Pool Name
RESOURCE_ POOL_ ID	Dynamic Resource Pool Number
RESOURCE_ POOl_ PRIORITY	Dynamic Resource Pool Priority
WAITING_ TIME	Task waiting time
RUNNING_ TIME	Task runtime
LOCK	Cluster lock
WAIT	Display mutexes that have been added and need to wait
INFO	Executed SQL statements
TRACE	TRACE information

5.7.1.2.19 GNODES_ TASK_ INFORMATION

Function Description

Record the resource usage information of SQL statements on data nodes.

Table Structure Description

Table 5224 Table Structure Information Explanation:

Column Name	Description
NODE_ NAME	Data node name
ID	Session ID
TASKID	Task ID
SUBTASKID	Subtask ID
THREADID	Thread ID
USER	Execute User
HOST	Execution node
DB	Affiliated database
COMMAND	SQL Type
START_ TIME	Start execution time
TIME	execution time

STATE	SQL Execution Status
RESOURCE_ POOL_ NAME	Dynamic Resource Pool Name
RESOURCE_ POOL_ ID	Dynamic Resource Pool Number
RESOURCE_ POOl_ PRIORITY	Dynamic Resource Pool Priority
WAITING_ TIME	Task waiting time
RUNNING_ TIME	Task runtime
PARALLEL_ DEGREE	Parallelism
CPU_ USAGE	CPU usage
MEM_ USAGE	Memory usage
TEMP_ DISKSPACE_ SORT	SORT Temporary Disk Space Usage
TEMP_ DISKSPACE_ JOIN	JOIN Temporary Disk Space Usage
TEMP_ DISKSPACE_ AGGR	AGGR Temporary Disk Space Usage
INFO	Executed SQL statements
TRACE	TRACE information

5.7.1.2.20 COORDINATORS_ RESOURCE_ POOL_ USAGE

Function Description

Record the real-time usage of resource pools for all coordinator nodes, including the number of waiting tasks, the number of running tasks, etc.

Table Structure Description

Table -5225 Table Structure Information Explanation:

Column Name	Description
NODE_ NAME	Node Name
RESOURCE_ POOL_ ID	Resource pool ID, with the same value range as the field type value range
RESOURCE_ POOL_ NAME	Resource Pool Name
PRIORITY	Resource pool priority, value range [1,8]
WAITING_ TASKS	The current number of tasks waiting for this resource pool within the cluster, with the same value range as the field type
RUNNING_ TASKS	The number of tasks currently running in this resource pool within the cluster, with the same value range as the field type value range
VC_ ID	Virtual cluster ID

VC_ NAME	Virtual cluster name

5.7.1.2.21 GNODES_ RESOURCE_ POOL_ USAGE

Function Description

Record the real-time usage of resource pools for data nodes, including CPU, memory, and disk usage information.

Table Structure Description

Table 5226 Table Structure Information Explanation:

Column Name	Description
NODE_ NAME	Node Name
VC_ ID	Virtual cluster ID
VC_ NAME	Virtual cluster name
RESOURCE_ POOL_ ID	Resource pool ID, with the same value range as the field type value range
RESOURCE_ POOL_ NAME	Resource Pool Name
PRIORITY	Resource pool priority, value range [1,8]
RUNNING_ TASKS	The current number of tasks waiting for this resource pool within the cluster, with the same value range as the field type
WAITING_ TASKS	The number of tasks currently running in this resource pool within the cluster, with the same value range as the field type value range
CPU_ USAGE	CPU usage rate (same as the top command)
MEM_ USAGE	Memory usage, with the same value range as the field type, in bytes
DISK_ USAGE	Disk usage, with the same value range as the field type, in bytes
DISK_ WRITEIO	Disk write IO bandwidth usage, with the same value range as the field type, in bytes/s
DISK_ READIO	Disk read IO bandwidth usage, with the same value range as the field type, in bytes/s

5.7.1.2.22 GNODES_ USER_ DISKSPACE_ USAGE

Function Description

Record the usage of user disk quotas, including user limited disk space, user actual usage of disk space, etc.

Table Structure Description

Table 5227 Table Structure Information Explanation:

Column Name	Description
NODE_ NAME	Node Name
User	User Name
user_ limit_ storage_ size	User disk space limit quota
user_ storage_ size	User disk space usage

5.7.1.2.23 COORDINATORS_ RESOURCE_ POOL_ STATUS

Function Description

Record the historical usage of resource pools in the cluster layer, including the number of running tasks, average task running time, and average task waiting time.

Table Structure Description

Table 5228 Table Structure Information Explanation:

Column Name	Description
NODE_ NAME	Node Name
RESOURCE_ POOL_ ID	Resource pool ID number, with the same value range as the field type value range
RESOURCE_ POOL_ NAME	Resource Pool Name
PRIORITY	Resource pool priority, value range [1,8]
SERVIED_ TASKS	The number of tasks executed by this resource pool within the cluster, with the same value range as the field type
WAITING_ AVG_ TIME	The average waiting time of tasks in this resource pool within the cluster, with values ranging from the same field type, in seconds
RUNNING_ AVG_ TIME	The average running time of tasks in this resource pool within the cluster, with values ranging from the same field type, in seconds
SAMPLE_ TIME	Sampling time point
VC_ ID	Virtual cluster ID
VC_ NAME	Virtual cluster name

5.7.1.2.24 GNODES_ RESOURCE_ POOL_ STATUS

Function Description

Record the historical usage of resource pools for data nodes, including CPU, memory,

and disk usage information.

Table Structure Description

Table 5229 Table Structure Information Explanation:

Column Name	Description
NODE_ NAME	Node Name
VC_ ID	Virtual cluster ID
VC_ NAME	Virtual cluster name
RESOURCE_ POOL_ ID	Resource pool ID number, with the same value range as the field type value range
RESOURCE_ POOL_ NAME	Resource Pool Name
PRIORITY	Resource pool priority, value range [1,8]
WAITING_ TASKS	The current number of tasks waiting for this resource pool within the cluster, with the same value range as the field type
RUNNING_ TASKS	The number of tasks currently running in this resource pool within the cluster, with the same value range as the field type value range
CPU_ USAGE	CPU usage rate (same as the top command)
MEM_ USAGE	Memory usage, with the same value range as the field type, in bytes
DISK_ USAGE	Disk usage, with the same value range as the field type, in bytes
DISK_ WRITEIO	Disk write IO bandwidth usage, with the same value range as the field type, in bytes/s
DISK_ READIO	Disk read IO bandwidth usage, with the same value range as the field type, in bytes/s
SAMPLE_ TIME	Sampling time point

5.7.1.2.25 RESOURCE_ POOL_ EVENTS

Function Description

Record cluster level resource pool event information.

Table Structure Description

Table 5230 Table Structure Information Explanation:

Column Name	Description
NODE_ NAME	Node Name
VC_ ID	Virtual cluster ID
VC_ NAME	Virtual cluster name

RESOURCE_ POOL_ ID	Resource pool ID number, with the same value range as the field type value range
RESOURCE_ POOL_ NAME	Resource Pool Name
EVENT_ TIME	Event occurrence time
TASK_ ID	Task ID
STATEMENT	SQL statement
EVENT_ TYPE	Event types, including terminate (terminate tasks that have already started executing) and reject (terminate tasks that have not yet started executing)
EVENT_ DESCRIPTION	Reason description for event occurrence, wait timeout, run timeout, insufficient resources, etc

5.7.1.2.26 COORDINATOR_ RESOURCE_ POOL_ USAGE

Function Description

Record the real-time usage of the resource pool of the current coordinator node, including the number of waiting tasks, the number of running tasks, etc.

Table Structure Description

Table 5231 Table Structure Information Explanation:

Column Name	Description
RESOURCE_ POOL_ ID	Resource Pool ID
RESOURCE_ POOL_ NAME	Resource Pool Name
PRIORITY	Resource pool priority, value range [1,8]
WAITING_ TASKS	The current number of tasks waiting for this resource pool within the cluster, with the same value range as the field type
RUNNING_ TASKS	The number of tasks currently running in this resource pool within the cluster, with the same value range as the field type value range
VC_ ID	Virtual cluster ID
VC_ NAME	Virtual cluster name

5.7.1.2.27 SCHEMATA

Function Description

Provide information about all databases in the current instance, where the results of show databases are obtained.

Table Structure Description

Table 5232 Table Structure Information Explanation:

Column Name	Description
CATALOG_ NAME	Register directory name
VC_ NAME	Name of the virtual cluster to which it belongs
SCHEMA_ NAME	Database name
DEFAULT_ CHARACTER_ SET_ NAME	Default Character Set Name
DEFAULT_ COLLATION_ NAME	Default Proofreading Name
SQL_ PATH	NULL
MIRROR_ VC_ NAME	Mirror virtual cluster name, empty if not present

5.7.1.2.28 SCHEMA_ PRIVILEGES

Function Description

The database limit table provides information about database permissions.

Table Structure Description

Table 5233 Table Structure Information Explanation:

Column Name	Description
GRANTEE	Permission Owner
TABLE_ CATALOG	Table of Contents for Registration
TABLE_ VC	Affiliated VC
TABLE_ SCHEMA	Affiliated database
PRIVILEGE_ TYPE	Permission Type
IS_ GRANTABLE	Do you have the authority to grant permissions

5.7.1.2.29 SESSION_ STATUS

Function Description

The status information of the current session.

Table Structure Description

Table 5234 Table Structure Information Explanation:

Column Name	Description
VARIABLE_ NAME	Variable name
VARIABLE_ VALUE	Variable value
SESSION_ LEVEL	Is it a session level state
WRITEABLE	Can changes be made during service operation

5.7.1.2.30 SESSION_ VARIABLES

Function Description

The variable name and value of the current session.

Table Structure Description

Table 5235 Table Structure Information Explanation:

Column Name	Description
VARIABLE_ NAME	Variable name
VARIABLE_ VALUE	Variable value
SESSION_ LEVEL	Is it a session level variable
WRITEABLE	Can changes be made during service operation

5.7.1.2.31 STATISTICS

Function Description

Provide information about table indexes.

Table Structure Description

Table 5236 Table Structure Information Explanation:

Column Name	Description
TABLE_ CATALOG	Data Table Registration Directory
TABLE_ VC	The virtual cluster name where the database of the table to which the index belongs is located
TABLE_ SCHEMA	The database name of the table to which the index belongs
TABLE_ NAME	The table name to which the index belongs
NON_ UNIQUE	If the index cannot include duplicate words, it is 0; if possible, it is 1
INDEX_ SCHEMA	The database name to which the index belongs (usually the same as table_schema)
INDEX_ NAME	Index Name
SEQ_ IN_ INDEX	The serial number of the column in the index, starting from 1
COLUMN_ NAME	Column names for indexes
COLLATION	How are columns stored in the index, [A (ascending) | NULL (unclassified)]
CARDINALITY	An estimate of the number of unique values in an index
SUB_ PART	If the column is only partially indexed, it is the number of indexed columns; if the entire column is indexed, it is NULL
PACKED	How keywords are compressed, default to NULL

NULLABLE	Is it empty
INDEX_ TYPE	The type of index, [BTREE | PRIMARY | FULL TEXT | HASH | RTREE]
COMMENT	Annotations and remarks for the index

5.7.1.2.32 TABLES

Function Description

Provides information about tables in the database (including views), detailing which library the table belongs to, table type, table engine, creation time, and other information.

Table Structure Description

Table 5237 Table Structure Information Explanation:

Column Name	Description
TABLE_ CATALOG	Table of Contents for Registration
TABLE_ VC	The virtual cluster name of the database to which the table belongs
TABLE_ SCHEMA	Database name to which the table belongs
TABLE_ NAME	Table Name
TABLE_ TYPE	Table type [VIEW | BASE TABLE]
ENGINE	Database engine used
VERSION	Version, default 0
ROW_ FORMAT	Row format [Compact | Dynamic | Fixed]
TABLE_ ROWS	How many rows of data are there in the table
AVG_ ROW_ LENGTH	Average row length
DATA_ LENGTH	Data length
MAX_ DATA_ LENGTH	Maximum data length
INDEX_ LENGTH	Index length
DATA_ FREE	How much space is left
AUTO_ INCREMENT	Autoincrement and current value for autoincrement primary keys
CREATE_ TIME	Table creation time
UPDATE_ TIME	Table update time
CHECK_ TIME	Check time of the table
TABLE_ COLLATION	Character Verification Encoding Set for Tables
CHECKSUM	Checksum
CREATE_ OPTIONS	Create Options
TABLE_ LIMIT_ STORAGE_ SIZE	Table limit size (the storage size of the current table cannot exceed this value, 0 indicates no limit)
TABLE_ STORAGE_ SIZE	Table storage size
TABLE_ DATA_ SIZE	Table data section (under the systable space

	directory) size
TABLE_ COMMENT	Notes and remarks on the table
LOCAL_ HASH_ INDEX_ FILE_ SIZE	Local hash file size
GLOBAL_ HASH_ INDEX_ FILE_ SIZE	Global hash file size
SCN	system change number
TABLE_ ID	Table ID
OWNER_ UID	The ID of the table owner
VC_ ID	Virtual cluster ID to which it belongs
TABLESPACE_ NAME	Tablespace name
TABLESPACE_ PATH	Table space storage path

5.7.1.2.33 TABLE_ CONSTRAINTS

Function Description

Describing the tables with constraints and the constraint types of the tables.

Table Structure Description

Table 5238 Table Structure Information Explanation:

Column Name	Description
CONSTRAINT_ CATALOG	Constraint Registration Directory
CONSTRAINT_ SCHEMA	The database name to which the constraint belongs
CONSTRAINT_ NAME	Name of constraint
TABLE_ VC	The virtual cluster name where the constraint belongs to the table
TABLE_ SCHEMA	The database name to which the constraint dependency table belongs (usually the same as constraint_schema)
TABLE_ NAME	The table name to which the constraint belongs
CONSTRAINT_ TYPE	Constraint type [primary key | foreign key | unique | check]

5.7.1.2.34 TABLE_ PRIVILEGES

Function Description

Record the operation permission information for the table.

Table Structure Description

Table 5239 Table Structure Information Explanation:

Column Name	Description
GRANTEE	Permission Owner
TABLE_ CATALOG	Data Table Registration Directory
TABLE_ VC	Name of the virtual cluster to which it belongs
TABLE_ SCHEMA	Database name
TABLE_ NAME	Table name to which it belongs
PRIVILEGE_ TYPE	Permission Type
IS_ GRANTABLE	Do you have the authority to grant permissions

5.7.1.2.35 TRIGGERS

Function Description

Event triggers provide information about the triggering program and must have super permission to view the table.

Table Structure Description

Table -5240 Table Structure Information Explanation:

Column Name	Description
TRIGGER_ CATALOG	Trigger registration directory, NULL
TRIGGER_ VC	Virtual cluster name where the trigger is located
TRIGGER_ SCHEMA	The database name where the trigger is located
TRIGGER_ NAME	Trigger Name
EVENT_ MANIPULATION	Trigger event type [Insert | DELETE | UPDATE]
EVENT_ OBJECT_ CATALOG	Trigger related table registration directory, NULL
EVENT_ OBJECT_ SCHEMA	The database where the trigger related table is located
EVENT_ OBJECT_ TABLE	Trigger related table name
ACTION_ ORDER	Triggers in the same table have all similar trigger order positions, always 0, because the same table cannot have more than one trigger with the same EVENT_ MANIPULATION and ACTION_ TIMING's trigger
ACTION_ CONDITION	Always NULL
ACTION_ STATEMENT	The trigger body, i.e. the execution state when the trigger is triggered, is encoded using UTF-8
ACTION_ ORIENTATION	The value is always' ROW '
ACTION_ TIMING	Is the trigger triggered before or after the event is triggered, [Before | After]
ACTION_ REFERENCE_ OLD_ TABLE	Always NULL
ACTION_ REFERENCE_ NEW_	Always NULL

TABLE	
ACTION_ REFERENCE_ OLD_ ROW	Old Column Discriminator [OLD]
ACTION_ REFERENCE_ NEW_ ROW	New Column Discriminator [NEW]
CREATED	Creation time
SQL_ MODE	Execution mode of SQL
DEFINER	User who created the trigger
CHARACTER_ SET_ CLIENT	The character set used by the session during creation
COLLATION_ CONNECTION	The validation set used by the session during creation
DATABASE_ COLLATION	Verification set of related databases

5.7.1.2.36 USER_ PRIVILEGES

Function Description

The user permission table provides information about global permissions.

Table Structure Description

Table 5-241 Table Structure Information Explanation:

Column Name	Description
GRANTEE	Permission Owner
TABLE_ CATALOG	Register directory name, NULL
PRIVILEGE_ TYPE	Permission Type
IS_ GRANTABLE	Do you have the authority to grant permissions

5.7.1.2.37 LOAD_ STATUS

Function Description

Record information on loading progress.

Table Structure Description

Table 5-242 Table Structure Information Explanation:

Column Name	Description
SCN	system change number
VC_ NAME	Virtual cluster name
DB_ NAME	Database name
TB_ NAME	Table Name

IP	Loader IP
STATE	Load Status
START_ TIME	Load start time
ELAPSED_ TIME	Load End Time
AVG_ SPEED	Loading speed
PROGRESS	Loading progress
TOTAL_ SIZE	Total file length
LOADED_ SIZE	Amount of data loaded
LOADED_ RECORDS	Number of loaded data entries
SKIPPED_ RECORDS	Number of skipped data entries
DATA_ SOURCE	data source
SQL_ CMD	SQL for loading tasks

5.7.1.2.38 LOAD_ TRACE

Function Description

Record the loading error log information of the current coordinator node.

Table Structure Description

Table 5243 Table Structure Information Explanation:

Column Name	Description
TASK_ ID	Load ID
DB_ NAME	Load Database Name
TB_ NAME	Load Table Name
ERR_ DATA_ IP	Node IP that generated incorrect data
FILE_ NAME	Load File Name
FILE_ OFFSET	Error data offset
RECORD_ LEN	Incorrect Data President
ERR_ COLUMN	Error data column number
ERR_ REASON	Specific reasons for incorrect data
ERR_ DATA	Incorrect data

5.7.1.2.39 CLUSTER_ LOAD_ TRACE

Function Description

Record the loading error log information of all coordinator nodes.

Table Structure Description

Table 5244 Table Structure Information Explanation:

Column Name	Description
TASK_ ID	Load ID
DB_ NAME	Load Database Name
TB_ NAME	Load Table Name
ERR_ DATA_ IP	Node IP that generated incorrect data
FILE_ NAME	Load File Name
FILE_ OFFSET	Error data offset
RECORD_ LEN	Incorrect Data President
ERR_ COLUMN	Error data column number
ERR_ REASON	Specific reasons for incorrect data
ERR_ DATA	Incorrect data

5.7.1.2.40 LOAD_ RESULT

Function Description

Record the loading result information of the current coordinator node.

Table Structure Description

Table 5245 Table Structure Information Explanation:

Column Name	Description
TASK_ ID	Load ID
DB_ NAME	Load Database Name
TB_ NAME	Load Table Name
USER	Current loaded username
ACCESS_ IP	Load origin IP
HOST_ IP	Client IP
START_ TIME	Load start time
END_ TIME	Load End Time
ELAPSED_ TIME	Loading time
TOTAL_ SIZE	Total size of loaded files
AVERAGE_ SPEED	Average loading speed
LOADED_ RECORDS	Number of loaded data entries
SKIPPED_ RECORDS	Skipped number of loading data
IGNORED_ FILES	Number of skipped files loaded
RESULT	Load Results
SQL_ CMD	Load SQL
MESSAGE	error message

5.7.1.2.41 CLUSTER_ LOAD_ RESULT

Function Description

Record the loading result information of all coordinator nodes.

Table Structure Description

Table 5246 Table Structure Information Explanation:

Column Name	Description
TASK_ ID	Load ID
DB_ NAME	Load Database Name
TB_ NAME	Load Table Name
USER	Current loaded username
ACCESS_ IP	Load origin IP
HOST_ IP	Client IP
START_ TIME	Load start time
END_ TIME	Load End Time
ELAPSED_ TIME	Loading time
TOTAL_ SIZE	Total size of loaded files
AVERAGE_ SPEED	Average loading speed
LOADED_ RECORDS	Number of loaded data entries
SKIPPED_ RECORDS	Skipped number of loading data
IGNORED_ FILES	Number of skipped files loaded
RESULT	Load Results
SQL_ CMD	Load SQL
MESSAGE	error message

5.7.1.2.42 VIEWS

Function Description

Information about views in the database is provided, and permission to show views is required, otherwise view information cannot be viewed.

Table Structure Description

Table 5247 Table Structure Information Explanation:

Column Name	Description
TABLE_ CATALOG	The registration directory of the view, with all values being NULL

TABLE_ VC	Virtual cluster name where the view is located
TABLE_ SCHEMA	The library name where the view is located
TABLE_ NAME	View Name
VIEW_ DEFINITION	Definition of Views
CHECK_ OPTION	Possible values include: [NONE | CASCADE | LOCAL]. 8a does not support view addition, deletion, or modification, this parameter can be ignored
IS_ UPDATABLE	Can it be updated
DEFINER	The creator of the view, in the format ' user_name@host_name '
SECURITY_ TYPE	Security type
CHARACTER_ SET_ CLIENT	The character set used by the session during creation
COLLATION_ CONNECTION	Verification set used for connection creation

5.7.1.2.43 VC

Function Description

Record virtual cluster information.

Table Structure Description

Table -5248 Table Structure Information Explanation:

Column Name	Description
ID	Virtual cluster number
NAME	Virtual cluster name
DEFAULT	Is it the default virtual cluster

5.7.1.2.44 CACHES

Function Description

Record cache information.

Table Structure Description

Table -5249 Table Structure Information Explanation:

Column Name	Description
VC	Virtual cluster name
DATABASE	Database Name
TABLE	Table Name

5.7.1.2.45 PRIORITIES

Function Description

Priority status display (related to cgroups and priority queues).

Table Structure Description

Explanation of Table -5250 Structure Information:

Column Name	Description
NODE_ NAME	Node Name
GROUP	Resource Group Number
PRIORITY	Priority Number
PRIORITY_ WEIGHT	Priority counterweight
STATUS	Control Group Status
DESCRIPTION	Control group parameter description

5.7.1.2.46 TABLE_ LOCKS

Function Description

Used to display the current usage of table locks.

Table Structure Description

Explanation of Table -5251 Structure Information:

Column Name	Description
VC	Virtual cluster name
DB	Database Name
TableName	Table Name
InsertFlag	Tag for Insert operation
CommitFlag	Markups for Commint operations
RDLCKs	Number of read locks held by the table
WRLCKs	Number of write locks held by the table
RecWRLCKs	Number of data receiving locks held by the table
ExLCK	Does the table hold an exclusive lock? 1 table does, 0 indicates none
ExWRLCK	Does the table hold exclusive write locks? 1 indicates yes, 0 indicates no
Sessions	Sessions Information

5.7.1.2.47 MEMORY_ HEAP_ INFO

Function Description

Information about each heap in memory.

Table Structure Description

Explanation of Table 5252 Structure Information:

Column Name	Description
HEAP	Heap Name
TOTAL_ SIZE	Total size
USED_ SIZE	Used size
AVAILABLE_ SIZE	Available size
USE_ MALLOC_ SIZE	Additional application size

5.7.1.2.48 CACHE_ ACCESS_ INFO

Function Description

The memory access status of the select/insert/update operation, the value in the delete/show operation table remains unchanged, and the value in the gbased restart table is reset to zero.

Table Structure Description

Table 5253 Table Structure Information Explanation:

Column Name	Description
ACCESS_ TIMES	The total number of memory accesses since GBased startup
HIT_ TIMES	Total number of hits
MISS_ TIMES	Total number of misses
HIT_ RATE	hit rate

5.7.1.2.49 CLUSTER_ TABLES

Function Description

Record the disk space information occupied by the table. When querying from this system table, table must be specified_ Schema and table_ Name, cannot blur queries and summaries.

Table Structure Description

Explanation of Table 5254 Structure Information:

Column Name	Description
TABLE_ VC	The virtual cluster code of the table to be queried
TABLE_ SCHEMA	The database name to which the table to be queried belongs
TABLE_ NAME	Table name of the table to be queried
CREATE_ TIME	Creation time of the table to be queried
UPDATE_ TIME	Last update time of the table to be queried
TABLE_ DATA_ SIZE	Data occupies space
TABLE_ STORAGE_ SIZE	Total space occupied by the table
TABLE_ TYPE	Table type: replicated - replicated table; Random - Random distribution table; Ha'sh - Hash distribution table;

5.7.1.2.50 CLUSTER_ TABLE_ SEGMENTS

Function Description

Record the disk space occupied by each shard's data.

Table Structure Description

Explanation of Table -5255 Structure Information:

Column Name	Description
TABLE_ VC	The virtual cluster code of the table to be queried
TABLE_ SCHEMA	The database name to which the table to be queried belongs
TABLE_ NAME	Table name of the table to be queried
SUFFIX	Partition name
HOST	Node IP
TABLE_ DATA_ SIZE	The storage space occupied by the pure data of this partition
TABLE_ STORAGE_ SIZE	The storage space occupied by the pure data and metadata of this shard
DATA_ PERCENT	The proportion of pure data in all shards of this shard

5.7.1.2.51 KAFKA_ CONSUMER_ STATUS

Function Description

Record the running status information of transactional consumer tasks.

Table Structure Description

Explanation of Table -5256 Structure Information:

Column Name	Description
CONSUMER	Consumer Name

IP	IP of the Consumer task running node
TOPIC	Topic name of consumption data
STATUS	Running status. Consumer task: current status Start: Executing Stop: Stopped Waiting start: Starting
MIN_ OFFSET	The smallest offset in the Topic queue
MAX_ OFFSET	The maximum offset in the topic queue
CUR_ OFFSET	The offset of the currently parsed message
PROCESS_ OFFSET	Offsets being handed over to the DML layer for processing
COMMIT_ OFFSET	Last submitted offset
OP_ TS	The timestamp of the last submitted data
EXCEPTION	If there are currently exceptions, display exception information (such as JSON parsing errors, target table does not exist, etc.), otherwise it is empty

5.7.1.2.52 KAFKA_ CONSUMER_ EFFICIENCY

Function Description

Record the operational efficiency of transactional consumer tasks, including time consumption statistics for each stage.

Table Structure Description

Explanation of Table -5257 Structure Information:

Column Name	Description
NAME	Consumer Name
EMPTY	Time consumption of kafka local cache queue being empty
CONSUME	The time required to retrieve data from the Kafka local cache queue
PARSE	Time consumption for parsing kafka messages
EXE_ SQL	The time required to execute SQL statements
COMMIT	Submission time
SLEEP	Time spent with an empty buffer
INSERT	Time consumption of Insert operation
UPDATE	Time consumption of update operation
DELETE	The time consumption of the Delete operation

5.7.1.2.53 KAFKA_ LOADER_ CONSUMER_ STATUS

Function Description

Record the running status of the loaded consumer task.

Table Structure Description

Explanation of Table 5258 Structure Information:

Column Name	Description
CONSUMER	Consumer Name
IP	The IP of the coordinator node where the consumer is running
TOPIC	Topic name of consumption data
STATUS	Current running status of Consumer task Start: Executing Stop: Stopped
PARTITION_ OFFSETS	The offset of the last landing data, including the offsets of each partition
EXCEPTION	If there are currently exceptions, display exception information (such as JSON parsing errors, target table does not exist, etc.), otherwise it is empty

5.7.1.2.54 SYS_ SCN

Function Description

System change number, the system change number.

Table Structure Description

Explanation of Table 5259 Structure Information:

Column Name	Description
SCN	System change number, this number has changed after the database update

5.7.1.2.55 DML_ INFO

Function Description

Record the number of running dml statements.

Table Structure Description

Table 5260 Table Structure Information Explanation:

Column Name	Description
COUNT	The fields inside refer to the number of DML statements currently running in the library

5.7.1.2.56 ALL_ ENCRYPTION_ CERTIFICATE_ STATUS

Function Description

Record the current encryption status information of all nodes.

Table Structure Description

Table 5261 Table Structure Information Explanation:

Column Name	Description
NODE_ NAME	Node Name
IS_ CREATE	Do you want to create a key
KEY_ TYPE	Key type (0 plaintext. 1 ciphertext)
OPEN_ STATUS	Key open status (ON on, OFF off)

5.7.1.2.57 COORDINATOR_ ENCRYPTION_ CERTIFICATE_ STATUS

Function Description

Record the current encryption status information of the cluster layer.

Table Structure Description

Table 5262 Table Structure Information Explanation:

Column Name	Description
COORDINATOR_ NAME	Node Name
IS_ CREATE	Do you want to create a key
KEY_ TYPE	Key type (0 plaintext. 1 ciphertext)
OPEN_ STATUS	Key open status (ON on, OFF off)

5.7.1.2.58 GNODE_ ENCRYPTION_ CERTIFICATE_ STATUS

Function Description

Record the current encryption status information of all node points.

Table Structure Description

Table 5263 Table Structure Information Explanation:

Column Name	Description
GNODE_ NAME	Node Name
IS_ CREATE	Do you want to create a key
KEY_ TYPE	Key type (0 plaintext. 1 ciphertext)
OPEN_ STATUS	Key open status (ON on, OFF off)

5.7.1.2.59 ENCRYPTION_ CERTIFICATE_ STATUS

Function Description

Record the current encryption status information of this node.

Table Structure Description

Table 5264 Table Structure Information Explanation:

Column Name	Description
HOST	Node Name
IS_ CREATE	Do you want to create a key
KEY_ TYPE	Key type (0 plaintext. 1 ciphertext)
OPEN_ STATUS	Key open status (ON on, OFF off)

5.7.1.2.60 TABLESPACES

Function Description

Record tablespace related information.

Table Structure Description

Table 5265 Table Structure Information Explanation:

Column Name	Description
NODE_ NAME	Node Name
DB_ NAME	Database Name
TABLESPACE_ ID	Table space encoding
TABLESPACE_ NAME	Tablespace name
TABLESPACE_ PATH	Table space path
IS_ DEFAULT	Is it the default tablespace
MAX_ SIZE	Maximum available space size
USED_ SIZE	Used Space Size

FREE_ SIZE	Available Space Size
SEG_ SIZE	SEG file splitting size of the table
VC_ NAME	Name of VC

5.7.1.2.61 TABLESPACE_ NAMES

Function Description

Record the name, path, and other related information of the tablespace.

Table Structure Description

Table 5266 Table Structure Information Explanation:

Column Name	Description
TABLESPACE_ NAME	Tablespace name
TABLESPACE_ PATH	Table space path
IS_ DEFAULT	Is it the default tablespace

5.7.1.2.62 CLUSTER_ MONIT_ INFO

Function Description

Obtain cluster status information, memory information, and disk information.

Table Structure Description

Table 5321 Table Structure Information Explanation:

Column Name	Description
HOST	Server IP
GCWARE_ STATE	GCWARE Service Status 1: OPEN 2: Close (abnormal) 0: This node does not include GCWARE services
GCLUSTER_ STATE	GCLUSTER Service Status 1: OPEN 2: Close (abnormal) 0: This node does not include GCLUSTER service
GNODE_ STATE	GNODE Service Status 1: OPEN 2: Close (abnormal)

	0: This node does not include GNODE services
SYNCSERVER_ STATE	SYNCSERVER service status 1: OPEN 2: Close (abnormal) 0: This node does not include SYNCSERVER service
COORSERVER_ DATA_ STATUS	The DataState state of the coordinator node 1: Normal 2: Abnormal 0: This node is not a coordinator node
DATASERVER_ DATA_ STATUS	The DataState state of the node 1: Normal 2: Abnormal 0: This node is not a node node
GCLUSTER_ VMSIZE	The size of virtual memory used by the gcluster process (GB), 0: This node does not contain the gcluster process
GCLUSTER_ VMRSS	The physical memory size used by the gclusterd process in GB, 0: This node does not include the gclusterd process
GCLUSTER_ DISK_ AVAIL_ SIZE	$GCLUSTER_ The remaining space on the mounted disk corresponding to the HOME/userdata/gcluster directory (GB), 0: This node does not contain the gclusterd process
GCLUSTER_ DISK_ AVAIL_ PERCENT	$GCLUSTER_ The percentage of usage of mounted disks corresponding to the HOME/userdata/gcluster directory, 0: This node does not contain gclusterd processes
GNODE_ VMSIZE	Size of virtual memory used by the gbased process (GB), 0: This node does not contain the gbased process
GNODE_ VMRSS	The physical memory size used by the gbased process in GB, 0: This node does not include the gbased process
GNODE_ DISK_ AVAIL_ SIZE	$GCLUSTER_ The remaining space on the mounted disk corresponding to the HOME/userdata/gbased directory (GB), 0: This node does not contain gbased processes
GNODE_ DISK_ AVAIL_ PERCENT	$GCLUSTER_ The percentage of usage of the mounted disk corresponding to the HOME/userdata/gbase directory, 0: This node

	does not contain gbased processes
CLUSTER_ STATUS	Cluster status 1: ACTIVE 2: LOCK (abnormal)
CLUSTER_ MODE	Cluster mode 1: NORMAL 2: READONLY 3: RECOVERY

5.7.1.2.63 SELECT_ STATISTIC

Function Description

information_ schema.SELECT_ STATISTIC checks the execution information of the currently executing SQL on all nodes.

Table Structure Description

field	meaning
ID	session id
HOST	Data Node IP
PORT	Data node port
NODE_ SID	Data Node Session ID
TIME	Data node task execution time
STATE	Data node task status: Start Task Start Send Query Task to Data Node UseResult Process the result set using Use Result StoreResult uses Store Result to process result sets ForwardResult uses Forward Result to process the result set FetchRow is executing fetch result set from Data node The HandleResultSelect executor is processing the result set Successfully executed the Finish task
SQL	Data Node Task SQL

5.7.2 GBase library

5.7.2.1 Table Function Definition

Table -5267 Table Function Information Description:

Table Name	Description
audit_ log	Record relevant information about operations performed by the database
audit_ policy	Record relevant information about the audit strategy.
cache_ access_ info	The memory access status of the select/insert/update operation, the delete/show operation table value does not change, and the gbased restart table value returns to zero
cluster_ resource_ pool_ usage_ history	Record the historical usage information of cluster layer resource pools
columns_ priv	Record column level authorization
consumer_ group	Record resource management consumption group information
consumer_ group_ user	Record the association relationship between resource management consumption groups and users
db	Record database level authorization information
db_ links	Record db_ Configuration information of link
event	Record relevant information about time triggers
func	Record UDF function related information
general_ log	The system table that records the executed SQL (can also be recorded in a file, controlled by the parameter log_output)
Host	Record host level authorization information
Memory_ heap_ info	Record memory usage information
New_ index_ conditions	
New_ index_ fields	
nodedatamap	Correspondence between storing hash key values and nodeid
Password_ history	Record user's password information
plugin	Plugins for recording databases
proc	Record stored procedure and function information

processlist	Record information about client connections
procs_ priv	Record authorization information for stored procedures and custom functions.
resource_ config	Record resource management configuration information
resource_ plan	Record resource plan information
resource_ plan_ directive	Record resource plan association information
resource_ pool	Record resource pool information information
resource_ pool_ events	Record resource pool event information
Role_ edges	Record user group management information
servers	
sql_ log	To record the log of SQL statements in the session, the parameter record needs to be configured_ From the code, it can be seen that only statements containing query operations will be recorded
Sql_ trace	
synonyms	Record synonym related information
table_ distribution	Current distribution information of the record table
tables_ priv	Record table level authorization information
Time_ zone	Record the mapping relationship between time zone ID and second hopping
Time_ zone_ leap_ second	Provide information on machine correction values for querying jump seconds
Time_ zone_ name	Provide a list of query time zone names and a mapping relationship between time zone IDs
Time_ zone_ transition	Provide jump second data for querying time zones
Time_ zone_ transition_ type	Provide specific second hopping information and corresponding data to the time zone for querying
user	Record the global user table, including information such as authorization, priority, and resource restrictions
user_ check	Record user security information
vcs_ priv	Record permission information at the virtual cluster level

5.7.2.2 Table Structure Description

5.7.2.2.1 audit_ log

Function Description

Record relevant information about the operations performed by the database.

Table Structure Description

Table 5268 Table Structure Information Explanation:

Column Name	Description
thread_ id	Thread number, the same as the ID in the processlist
taskid	Each SQL task number
start_ time	Task start time
end_ time	Task End Time
user_ host	Login username and IP, format: priv_ user[user]@hostname[ip]
uid	User UID
user	Distribute Task User
host_ ip	Host IP of the task user
query_ time	Task execution time
rows	Number of rows affected by task execution
vc_ id	Virtual cluster name to which the task belongs
db	The database name to which the task belongs
table_ list	List of tables involved in task execution, format: ` WRITE: `<db>``< tb>`; READ: `<db>`.`< tb>`; OTHER: `<db>`.`< tb>`;
sql_ text	Task execution time greater than long_ query_ SQL statement with time setting value
sql_ type	The type of SQL statement executed by the task, such as DDL, DML, DQL, others
sql_ command	Task classification for task execution
operators	Record the operators involved in the task. If multiple operators are involved, separate them with ','
status	SQL execution status, such as SUCCESS, FAILED, KILLED, etc
conn_ type	Task execution methods (CAPI, ODBC, JDBC, ADO.NET, STUDIO)

Table 5269 SQL_ Description of command value range:

INSERT	DELETE	UPDATE	LOAD
CREATE	CREATE DB	CREATE TABLE	CREATE VIEW

USER			
CREATE INDEX	CREATE PROCEDURE	CREATE FUNCTION	ALTER FUNCTION
ALTER EVENT	DROP USER	DROP DB	DROP TABLE
DROP VIEW	DROP INDEX	DROP PROCEDURE	DROP FUNCTION
DROP EVENT	TRUNCATE	GRANT	REVOKE
SELECT	OTHERS	RENAME_ USER	

Table 5270 Explanation of the Value Range of Operators:

START_ WITH	CONNECT_ BY	JOIN	WHERE	GROUP
OLAP_ GROUP	HAVING	OLAP_ FUNC	DISTINCT	ORDER
LIMIT				

5.7.2.2.2 audit_ policy

Function Description

Record relevant information about the audit strategy.

Table Structure Description

Table 5271 Table Structure Information Explanation:

Column Name	Description
Name	Audit Policy Name
Enable	Policy enable status, Y enable, N disable
Hosts	Restricted host name, empty indicates no restrictions
User	Restricted username, case sensitive, blank indicates no restrictions
Db	Restricted database name, empty indicates no restrictions
Obj_ type	Object types, including Table (VIEW): Table (View), Procedure: Stored Procedure, Function: Custom Function
Object	Match obj_ TPE specified object
Sql_ commands	Restricted commands such as insert, delete, etc
Long_ query_ time	Set the minimum value for recording SQL execution time

Status	Limit the execution result of the task, SUCCES: successful execution, FAILED: failed execution, and blank indicates no restriction

Table 5272 SQL_ Description of command value range:

INSERT	DELETE	UPDATE	LOAD
CREATE USER	CREATE DB	CREATE TABLE	CREATE VIEW
CREATE INDEX	CREATE PROCEDURE	CREATE FUNCTION	ALTER FUNCTION
ALTER EVENT	DROP USER	DROP DB	DROP TABLE
DROP VIEW	DROP INDEX	DROP PROCEDURE	DROP FUNCTION
DROP EVENT	TRUNCATE	GRANT	REVOKE
SELECT	OTHERS	RENAME_ USER	

5.7.2.2.3 cache_ access_ info

Function Description

The memory access status of the select/insert/update operation, the value in the delete/show operation table remains unchanged, and the value in the gbased restart table is reset to zero.

Table Structure Description

Table 5273 Table Structure Information Explanation:

Column Name	Description
ACCESS_ TIMES	The total number of memory accesses since GBased startup
HIT_ TIMES	Total number of hits
MISS_ TIMES	Total number of misses
HIT_ RATE	hit rate
SNOPSHOTTIME	Snapshot time

5.7.2.2.4 cluster_ resource_ pool_ usage_ history

Function Description

Record the historical usage of cluster layer resource pools.

Table Structure Description

Table -5274 Table Structure Information Explanation:

Column Name	Description
resource_ pool_ id	Resource Pool ID
resource_ pool_ name	Resource Pool Name
priority	Resource Pool Priority
serviced_ tasks	Number of tasks controlled by resource pool
waiting_ avg_ time	Task average waiting time
running_ avg_ time	Average task runtime
sampletime	Information recording time
vc_ id	Virtual cluster number
vc_ name	Virtual cluster name

5.7.2.2.5 columns_ priv

Function Description

Record column level authorization.

Table Structure Description

Table -5275 Table Structure Information Explanation:

Column Name	Description
Host	Column level permission host hostname
VC_ ID	Name of the virtual cluster to which it belongs
Db	The database to which the column belongs
User	User name for column level permissions
Table_ name	The table name to which the column belongs
Column_ name	Column names for column level permissions
Timestamp	Creation or update time of column level permissions
Column_ priv	Types of column level permissions

5.7.2.2.6 consumer_ group

Function Description

Record resource management consumption group information.

Table Structure Description

Table -5276 Table Structure Information Explanation:

Column Name	Description
consumer_ group_ id	Consumer Group ID

consumer_ group_ name	Consumer Group Name
comment	Annotation Information
vc_ id	Virtual cluster number

5.7.2.2.7 consumer_ group_ user

Function Description

Record the association relationship between resource management consumption groups and users.

Table Structure Description

Table 5277 Table Structure Information Explanation:

Column Name	Description
consumer_ group_ id	Consumer Group ID
user_ name	User Name

5.7.2.2.8 db

Function Description

Record database level authorization information using grant all on db_ name.* to user_ name。

Table Structure Description

Table 5278 Table Structure Information Explanation:

Column Name	Description
Host	Host hostname with database level permissions
VC_ ID	Virtual cluster number
Db	Database name for database level permissions
User	User name for database level permissions
Select_ priv	Allow use of select
Insert_ priv	Allow Insert
Update_ priv	Allow use of update
Delete_ priv	Allow delete
Create_ priv	Allow the use of create table
Drop_ priv	Allow drop
Drop_ table_ priv	Allow use of drop table
Drop_ view_ priv	Allow drop view
Drop_ database_ priv	Allow the use of drop database
Unmask_ priv	Allow unmask

Grant_ priv	Allow the use of grant
References_ priv	Placeholder for future functionality, currently useless
Index_ priv	Allow the use of create index and drop index
Alter_ priv	Allow use of alter table
Create_ tmp_ table_ priv	Allow the use of create temporary table
Lock_ tables_ priv	Allow the use of lock tables with select permission
Create_ view_ priv	Allow the use of create view
Show_ view_ priv	Allow use of show create view
Create_ routine_ priv	Allow the use of create stored procedures and functions
Alter_ routine_ priv	Allow alter stored procedures and functions
Execute_ priv	Allow the use of stored procedures and functions
Event_ priv	Allow the use of events
Trigger_ priv	Allow the use of triggers

5.7.2.2.9 db_ links

Function Description

Record db_ Configuration information for the link.

Table Structure Description

Table 5279 Table Structure Information Explanation:

Column Name	Description
owner	GBase users of link
db_ link	db_ Link Name
dblink_ priv	Shared or not
username	The username of the peer
password	Opposite Password
host	The host name of the gateway
created	Creation time

5.7.2.2.10 event

Function Description

Record relevant information about time triggers.

Table Structure Description

Table 5280 Table Structure Information Explanation:

Column Name	Description

vc_ id	Virtual cluster number
db	Database name
name	Event Name
body	Event Definition
definer	Event Creator
execute_ at	Event occurrence time
interval_ value	Execution interval
interval_ field	Unit of execution interval
created	Creation time
modified	Modification time
last_ executed	Last execution time
starts	Start execution time
ends	End Execution Time
status	Event status, available or not
on_ completion	Perform post completion actions
sql_ mode	Execution mode of SQL
comment	notes
execute_ mode	The execution mode of Event, where 0 indicates that the event is executed in isolate mode and 1 indicates that it is executed in exclusive mode, with a default value of 0.
time_ zone	time zone
character_ set_ client	Client character set
collation_ connection	Linking Character Set Comparison Method
db_ collation	Database Character Set Comparison Method
body_ utf8	Definition in utf8 format

5.7.2.2.11 **func**

Function Description

Record UDF function related information

Table Structure Description

Table -5281 Table Structure Information Explanation:

Column Name	Description
name	Function Name
ret	Return value type string | integer | real
dl	Dynamic connection library name
type	Is it a aggregate function
engine	Engine Name
instance	Instance Name
vcid	Virtual cluster number

5.7.2.2.12 general_ log

Function Description

Record the system table of the executed SQL (which can also be recorded in a file, subject to the parameter log_output).

Table Structure Description

Table 5282 Table Structure Information Explanation:

Column Name	Description
event_ time	The time when the event (including connect, quit, SQL, etc.) occurred
user_ host	User and host information for executing SQL
uid	
thread_ id	The thread corresponding to the session_ ID number
server_ id	The corresponding server ID number (usually 0) does not need to be considered
command_ type	Operation types, including Quit, Connect, Query, etc
argument	Specific information about the operation, including the executed SQL or the connected user information （ root@localhost ）

5.7.2.2.13 host

Function Description

Record host level authorization information.

Table Structure Description

Table 5283 Table Structure Information Explanation:

Column Name	Description
Host	The hostname used to access the database
VC_ ID	Virtual cluster number
Db	Database name
Select_ priv	Execute permissions for select statements
Insert_ priv	Insert permissions
Update_ priv	Permissions for update
Delete_ priv	Execution permissions for delete statements
Create_ priv	Permissions for creating a table
Drop_ priv	Permissions for drop table
Grant_ priv	Permissions for managing permissions
References_ priv	Placeholder for future functionality, currently useless
Index_ priv	Permissions for create index and drop index

Alter_ priv	Alter Permissions
Create_ tmp_ table_ priv	Create temporary table permissions
Lock_ tables_ priv	Allow the use of lock tables on tables with select permission
Create_ view_ priv	Permissions for creating views
Show_ view_ priv	Permission to use show create view
Create_ routine_ priv	Permissions for creating stored procedures and functions
Alter_ routine_ priv	Allow alter processes and functions
Event_ priv	Event permissions
Trigger_ priv	Trigger usage permissions

5.7.2.2.14 memory_ heap_ info

Function Description

Record memory usage information.

Table Structure Description

Table -5284 Table Structure Information Explanation:

Column Name	Description
HEAP	Heap Name
TOTAL_ SIZE	Total size
USED_ SIZE	Used size
AVAILABLE_ SIZE	Available size
USE_ MALLOC_ SIZE	Additional application size
SNOPSHOTTIME	Snapshot time

5.7.2.2.15 nodedatamap

Function Description

Record the correspondence between hash key values and nodeids. When calculating the hash of data, the hash key value is first calculated using the hash calculation formula (crc32()% 65536). Then, find the corresponding nodeid from the gbase. nodatamap. Finally, in the node information stored in GCWare, search for the corresponding data sharding information. When searching, the nodeid is the order in which the nodes are stored in GCWare.

Table Structure Description

Table -5285 Table Structure Information Explanation:

Column Name	Description
hashkey	Hash key value. Range 0-65535

nodeid	Table shard ID. Starting from 0, there is a one-to-one correspondence with shards. You can view the segment ID through gcadmin showdistribution. Here, nodeid=segment id -1
Data_ distribution_ id	The ID of the table data distribution information can be obtained through gcadmin showdistribution

5.7.2.2.16 password_ history

Function Description

Record the user's password information.

Table Structure Description

Table 5286 Table Structure Information Explanation:

Column Name	Description
Host	Login Host IP
User	user name
Password	password
Passwrod_ time	Password setting time

5.7.2.2.17 plugin

Function Description

Record the plugin information of the database.

Table Structure Description

Table 5287 Table Structure Information Explanation:

Column Name	Description
name	Plugin Name
dl	The name of the plugin dynamic link library, along with the plugin of the configuration file_ Dir usage

5.7.2.2.18 proc

Function Description

Record stored procedure and function information.

Table Structure Description

Table 5288 Table Structure Information Explanation:

Column Name	Description
Vc_ id	Virtual cluster name
db	Database name
name	Procedure and Function Names
type	Stored procedure or function type
specific_ name	Procedure and Function Names
language	The subprogram is composed of SQL statements, and the current supported language in the system is SQL, with SQL being its unique value
sql_ data_ access	Storage Procedure Data Access Characteristics Contains_ SQL: The subroutine does not contain statements to read or write data no_ SQL: The subroutine does not contain SQL statements reads_ SQL: The subroutine contains statements to read data, but not statements to write data data modifies_ sql_ Data: The subroutine contains statements for writing data
is_ deterministic	Certainty of output results Yes: The same input will result in the same output No: The same input may result in different outputs
security_ type	Specify who has permission to execute Invoker: Callers with permissions can execute Definer: Only the definer can execute
param_ list	parameter list
returns	return type
body	Procedure or function definition
definer	creator
created	Creation time
modified	Modification time
sql_ mode	Execution mode of SQL
comment	notes
character_ set_ client	Client character set
collation_ connection	Linking Character Set Comparison Method
db_ collation	Character Set Comparison Method for Database
body_ utf8	Definition in utf8 format

5.7.2.2.19 **processlist**

Function Description

Record information about client connections.

Table Structure Description

Table 5289 Table Structure Information Explanation:

Column Name	Description
ID	ID number of the processlist
USER	Connected Users
HOST	IP number: Port number
DB	The database being used by the process
COMMAND	The command executed by the current connection [Query | Connect | Sleep]
TIME	The duration of the current state
STATE	Display the execution status of SQL statements using the current connection
INFO	Show executing SQL statements
SNOPSHOTTIME	Snapshot creation time

5.7.2.2.20 procs_ priv

Function Description

Record authorization information for stored procedures and custom functions.

Table Structure Description

Table 5290 Table Structure Information Explanation:

Column Name	Description
Host	Hostname for process and function permissions
vc_ id	Virtual cluster name
Db	Database name for process and function permissions
User	User name for procedure and function permissions
Routine_ name	Procedure and Function Names
Routine_ type	Process and Function Categories
Grantor	Authorizer
Proc_ priv	Authorization Type
Timestamp	Creation and update time

5.7.2.2.21 resource_ config

Function Description

Record resource management configuration information.

Table Structure Description

Table 5291 Table Structure Information Explanation:

Column Name	Description
config_ name	Resource Management Configuration Item Name
config_ type	Resource Management Configuration Item Type (Number/String)
config_ int_ value	Resource Management Configuration Item Value (Numeric Type)
config_ str_ value	Resource Management Configuration Item Value (String Type)
Vc_ id	Virtual cluster name

5.7.2.2.22 resource_ plan

Function Description

Record resource plan information.

Table Structure Description

Table 5292 Table Structure Information Explanation:

Column Name	Description
resource_ plan_ id	Resource Plan ID
resource_ plan_ name	Resource Plan Name
comment	Annotated information for resource plans
vc_ id	Virtual cluster name

5.7.2.2.23 resource_ plan_ directive

Function Description

Record resource plan association information.

Table Structure Description

Table 5293 Table Structure Information Explanation:

Column Name	Description
resource_ plan_ directive_ name	Resource Plan Directive Name
resource_ plan_ id	Resource Plan ID
consumer_ group_ id	Consumer Group ID
resource_ pool_ id	Resource Pool ID
comments	Annotation Information
Vc_ Id	Virtual cluster name

5.7.2.2.24 resource_ pool

Function Description

Record resource pool information information.

Table Structure Description

Table 5294 Table Structure Information Explanation:

Column Name	Description
resource_ pool_ id	Resource Pool ID
resource_ pool_ name	Resource Pool Name
resource_ pool_ type	Resource Pool Type (Static/Dynamic)
parent_ resource_ pool_ id	Parent Resource Pool ID
priority	Resource Pool Priority
max_ memory	Memory usage limit
max_ tmp_ table_ space	Maximum disk temporary space usage
max_ disk_ space	Maximum disk space usage
task_ parallel	Task concurrency
max_ task_ number	Maximum number of tasks
cpu_ percent	CPU usage limit
disk_ write_ bps	Maximum IO usage for disk write operations
disk_ read_ bps	Maximum IO usage for disk read operations
waiting_ timeout	Maximum task waiting time
running_ timeout	Maximum task runtime
Vc_ id	Virtual cluster name

5.7.2.2.25 resource_ pool_ events

Function Description

Record resource pool event information.

Table Structure Description

Table 5295 Table Structure Information Explanation:

Column Name	Description
vc_ id	Virtual cluster number
vc_ name	Virtual cluster name
resource_ pool_ id	Resource Pool ID
resource_ pool_ name	Resource Pool Name
event_ time	Event occurrence time

task_ id	Task ID
statement	SQL statement
event_ type	Event type
event_ description	Event description information

5.7.2.2.26 role_ edges

Function Description

Record user group management information.

Table Structure Description

Table 5296 Table Structure Information Explanation:

Column Name	Description
FROM_ HOST	User Group IP
FROM_ USER	User group name
TO_ HOST	User IP
TO_ USER	user name
WITH_ ADMIN_ OPTION	Do you have the authority to grant user groups to other users Y: Permission to grant user groups to other users N: User groups cannot be granted permissions to other users

5.7.2.2.27 sql_ log

Function Description

To record the log of SQL statements in the session, the parameter record needs to be configured_ SQL. Only statements containing query operations will be recorded.

Table Structure Description

Table 5297 Table Structure Information Explanation:

Column Name	Description
MD5_ HASH	Md5hash value of query
COUNT	Number of times executed
SQLS	Query statement
DB	Database name
BEGIN_ TIME	start time
LAST_ TIME	Last execution time

5.7.2.2.28 synonyms

Function Description

Record information related to synonyms.

Table Structure Description

Table 5298 Table Structure Information Explanation:

Column Name	Description
synonym_ vcid	The virtual cluster number to which the synonym belongs
object_ vcid	The virtual cluster number to which the object belongs
type	SYNONYM: Synonym CACHE: Intelligent Cacha, which refers to the frequency of user heap cross domain remote table access, calculates the corresponding table, selects and creates a cache
owner	Empty indicates public synonyms, otherwise indicates the database name where private synonyms are located
synonym_ name	Synonym name
object_ owner	The library name where the object is located
object_ name	Object Name
db_ link	Dblink name, default to NULL

5.7.2.2.29 table_ distribution

Function Description

Record the current distribution information of the table.

Table Structure Description

Table 5299 Table Structure Information Explanation:

Column Name	Description
index_ name	database.table_ Name, used as the primary key
dbName	Database name to which the table belongs
tbName	Table Name
isReplicated	Is it a replicated table; Y - Yes; N - No
hash_ column	Data distribution columns of hash distribution tables
lmt_ storage_ size	Temporarily not used
table_ storage_ size	Temporarily not used
is_ nocopies	Is it a nocopies table? The current version does not support this type of table, and the column values are all NO
data_ distirbution_ id	ID of table data distribution information

vc_ id	Virtual cluster number
mirror_ vc_ id	Mirror virtual cluster number

5.7.2.2.30 tables_ priv

Function Description

Record table level authorization information.

Table Structure Description

Explanation of Table 5300 Table Structure Information:

Column Name	Description
Host	Host hostname for table level permissions
Vc_ id	Number of virtual cluster
Db	Database name for table level permissions
User	User name for table level permissions
Table_ name	Table name for table level permissions
Grantor	Authorizer of table level permissions
Timestamp	Creation or update time of table level permissions
Table_ priv	Table level permissions: 'Select','Insert','Update','Delete','Create','Drop','Grant','References','Index','Alter','Unmask','Create View','Show view','Trigger'
Column_ priv	Column level permissions: 'Select','Insert','Update','References'
Unmask_ priv	Allow unmask

5.7.2.2.31 Time_ zone

Function Description

Record the mapping relationship between time zone ID and jump seconds.

Table Structure Description

Explanation of Table 5301 Structure Information:

Column Name	Description
Time_ zone_ id	Time zone ID
Use_ leap_ seconds	Does the time zone use jump seconds

5.7.2.2.32 Time_ zone_ leap_ second

Function Description

Provide information on machine correction values for querying jump seconds.

Table Structure Description

Explanation of Table 5302 Structure Information:

Column Name	Description
Transition_ time	Transient time of second hopping
Correction	Correction value for jump seconds

5.7.2.2.33 Time_ zone_ name

Function Description

Provide a list of query time zone names and a mapping relationship between time zone IDs.

Table Structure Description

Explanation of Table 5303 Structure Information:

Column Name	Description
Name	Time zone name, which is time_ One of the valid values of the zone system variable
Time_ zone_ id	Time zone ID, which matches the table time_ The ID in the zone corresponds to

5.7.2.2.34 Time_ zone_ transition

Function Description

Provide jump second data for querying time zones.

Table Structure Description

Explanation of Table 5304 Structure Information:

Column Name	Description
Time_ zone_ id	Time zone ID
Transition_ time	And time_ zone_ leap_ Transition in the second table_ The time field has the same meaning
Transition_ type_ id	And time_ zone_ transition_ Transition in the type table_ type_

	ID corresponds to

5.7.2.2.35 Time_ zone_ transition_ type

Function Description

Provide specific second hopping information and corresponding data to the time zone for querying.

Table Structure Description

Explanation of Table 5305 Structure Information:

Column Name	Description
Time_ zone_ id	Time zone ID
Transition_ type_ id	And time_ zone_ Transition in the transition table_ type_ Corresponding ID value
Offset	Offset from UTC time
Is_ DST	Is it the Eastern Time Zone
Abbreviation	Abbreviation for standard time, such as GMT, which is time_ One of the valid values of the zone system variable

5.7.2.2.36 user

Function Description

Record global user table information, including authorization, priority, resource restrictions, and other information.

Table Structure Description

Explanation of Table 5306 Structure Information:

Column Name	Description
Host	The hostname used to access the database
User	User name used to access the database
Password	Password used to access the database
Default_ vc	Default virtual cluster name
Select_ priv	Execute permissions for select statements
Insert_ priv	Insert permissions
Update_ priv	Permissions for update
Delete_ priv	Execution permissions for delete statements
Create_ priv	Permissions for creating a table
Drop_ priv	Permissions for drop
Drop_ database_ priv	Permissions for drop database

Drop_ table_ priv	Permissions for drop table
Drop_ view_ priv	Permissions for drop view
Reload_ priv	Permission to use flush
Shutdown_ priv	Shutdown permission
Process_ priv	Permissions for show processlist
File_ priv	Using select Permissions for into file
Grant_ priv	Permissions for managing permissions
References_ priv	Placeholder for future functionality, currently useless
Index_ priv	Permissions for create index and drop index
Alter_ priv	Alter Permissions
Show_ db_ priv	Permission to show databases
Super_ priv	The user has super permissions, such as kill
Create_ tmp_ table_ priv	Create temporary table permissions
Lock_ tables_ priv	Allow the use of lock tables on tables with select permission
Execute_ priv	Permission to execute stored procedures and functions
Repl_ slave_ priv	Placeholder for future functionality, currently useless
Unmask_ priv	Dynamic desensitization permission (The original column name Repl_client_priv in the old version was changed to Unmask_priv for desensitization permission control due to obsolescence)
Create_ view_ priv	Permissions for creating views
Show_ view_ priv	Permission to use show create view
Create_ routine_ priv	Permissions for creating stored procedures and functions
Alter_ routine_ priv	Allow alter processes and functions
Create_ user_ priv	Create user, rename user, drop user, revoke all privileges, can only be used at the global level
Event_ priv	Event permissions
Trigger_ priv	Trigger usage permissions
ssl_ type	Support SSL standard encryption of secure fields
ssl_ cipher	Support SSL standard encryption of secure fields
x509_ issuer	Support x509 standard fields
x509_ subject	Support x509 standard fields
max_ questions	Maximum number of queries per hour for users
max_ updates	Maximum updates per hour for users
max_ connections	Maximum number of connections per hour for users
max_ user_ connections	Number of sessions simultaneously connected by users
max_ cpus	Maximum number of CPU cores used by users
max_ memories	Maximum memories used by users
max_ tmp_ space	Maximum temporary space used by users
resource_ group	User resource group, controlling CPU, memory, etc
task_ priority	user priority

user_ limit_ storage_ size	Storage capacity limit
user_ storage_ size	Current storage capacity
uid	User Code
plugin	User authentication method Gbase_ native_ Password: represents the username/password authentication method Kerberos: represents the kerberos authentication method
Auth_ string	Authentication information, the kerberos authentication method stores the client's principal, and the username/password authentication method is empty

5.7.2.2.37 user_ check

Function Description

Record user security information.

Table Structure Description

Explanation of Table -5307 Structure Information:

Column Name	Description
Host	Host IP
User	user name
attempt	Number of password retries
last_ attempt	The number of retries successfully logged in recently
locked	Is the user locked
password_ expired	Does the password expire
password_ last_ changed	Last password modification time
password_ life_ time	Password validity period, in days
password_ history	Password history list, ciphertext
host_ list	List of hosts allowed to log in
login_ time	Current login time
login_ host	This login to the host
last_ login_ time	Last login time
last_ login_ host	Recently logged in to the host
login_ count	Number of user logins

5.7.2.2.38 Vcs_ priv

Function Description

Record VC level permission information.

Table Structure Description

Explanation of Table -5308 Structure Information:

Column Name	Description
Host	Host IP
VC_ ID	Virtual cluster encoding
User	user name
Select_ priv	Execute permissions for select statements
Insert_ priv	Insert permissions
Update_ priv	Permissions for update
Delete_ priv	Execution permissions for delete statements
Create_ priv	Permissions for creating a table
Drop_ priv	Permissions for drop
Drop_ table_ priv	Permissions for drop table
Drop_ view_ priv	Permissions for drop view
Drop_ database_ priv	Permissions for drop database
Unmask_ priv	Unmask permissions
Grant_ priv	Permissions for managing permissions
References_ priv	Placeholder for future functionality, currently useless
Index_ priv	Permissions for create index and drop index
Alter_ priv	Alter Permissions
Create_ tmp_ table_ priv	Create temporary table permissions
Lock_ tables_ priv	Allow the use of lock tables on tables with select permission
Create_ view_ priv	Permissions for creating views
Show_ view_ priv	Permission to use show create view
Create_ routine_ priv	Permissions for creating stored procedures and functions
Alter_ routine_ priv	Allow alter processes and functions
Execute_ priv	Permission to execute stored procedures and functions
Event_ priv	Event permissions
Trigger_ priv	Trigger usage permissions

5.7.3 Gclusterdb library

5.7.3.1 Table Function Definition

Table -5309 Function Information Description:

Table Name	Description
dual	Internal virtual table
rebalancing_ status	Redistribution Status Information Table

5.7.3.2 Table Structure Description

5.7.3.2.1 dual

Function Description

Virtual tables, supporting the use of dual tables in distributed query plans.

Table Structure Description

Table 5310 Table Structure Information Explanation:

Column Name	Description
dummy	No practical significance

5.7.3.2.2 rebalancing_ status

Function Description

Redistribution status information table, used to view the progress of redistribution during the redistribution process.

Table Structure Description

Table 5311 Table Structure Information Explanation:

Column Name	Description
index_ name	Database name. Table name
db_ name	Database name
table_ name	Table Name
tmptable	The intermediate table name used during the Rebalance task execution. If an intermediate table is not used, then the field value is NULL.
start_ time	When in the STARTING state, indicates the time when the task was added. In the RUNNING state, start_ Time represents the time when the task was changed to RUNNING.
end_ time	Indicates the end time of the Rebalance task.
status	Represents the current state of the Rebalance task.
percentage	Indicates the execution progress of the Rebalance task.
priority	Indicates the priority of the Rebalance task. The lowest value is executed first.
host	Indicates which coordinator node executes the

	Rebalance task.
distribution_ id	Indicates which distribution ID the Rebalance task wants to Rebalance the table to.

5.7.3.2.3 Kafka_ consumers

Function Description

Record the information of the consumer.

Table Structure Description

Table 5312 Table Structure Information Explanation:

Column Name	Description
name	
type	
db	
table	
topic	
brokers	
partitions	
duration	
loader_ options	
status	
common_ options	

5.7.3.2.4 Sys_ sqls

Function Description

gclusterdb.sys_ SQL records the executed SQL information.

Table Structure Description

field	meaning
id	Cluster SQL unique identifier
isql	Cluster SQL
istop	Record the storage time

5.7.3.2.5 Sys_ sql_ elapsepernode

Function Description

gclusterdb.sys_ sql_ Elapsepernode records historical SQL execution information on all nodes.

Table Structure Description

field	meaning
id	Cluster SQL unique identifier
node	Data Node IP
node_ sid	Data node task session id
du	Execution time
errnum	1: Execution successful, 0 execution failed

5.7.4 performance_ Schema library

5.7.4.1 Table Function Definition

Table -5313 Table Function Information Description:

Table Name	Description
DISK_ USAGE_ INFO	Database disk space usage statistics for this node
CLUTER_ DISK_ USAGE_ INFO	Database disk space usage statistics for all nodes in the cluster
CACHE_ USAGE_ INFO	Database cache usage statistics for this node
CACHE_ CELL_ STATUS_ INFO	Statistical information of DC in database cache
HEAP_ USAGE_ INFO	Database heap usage statistics
SESSION_ MEMORY_ USAGE_ INFO	Session level memory statistics
MEMORY_ USAGE_ INFO	Memory section overview information
TABLES	Table Statistics
CLUSTER_ MONIT_ INFO	
MONIT_ INFO	

5.7.4.2 Table Structure Description

5.7.4.2.1 DISK_ USAGE_ INFO

Function Description

Database current node disk space usage statistics.

Table Structure Description

Table -5314 Table Structure Information Explanation:

Column Name	Description
HOST	Name of this node
DIR_ TYPE	Catalog type. ● Datadir: Data and metadata directory ● Logdir: Log directory ● gbase_ cache_ Data: During the SQL execution process, the data storage directory needs to be landed in the middle
PATH	DIR_ Absolute path of TYPE
DIR_ SIZE	Number of disk space occupied by the directory
FILESYSTEM	The device file path name corresponding to the file system where the directory is located, usually corresponding to the disk partition
SIZE	The available space capacity of FILESYTEM
USED	Space capacity already occupied
AVAIL	Available Space Capacity
PCT	Percentage of space usage

5.7.4.2.2 CLUTER_ DISK_ USAGE_ INFO

Function Description

Database current node disk space usage statistics.

Table Structure Description

Table -5315 Table Structure Information Explanation:

Column Name	Description
HOST	Node Name
DIR_ TYPE	Catalog Type Type Value Description: ● Datadir: Data and metadata directory ● Logdir: Log directory

	● gbase_ cache_ Data: During the SQL execution process, the data storage directory needs to be landed in the middle
PATH	DIR_ Absolute path of TYPE
DIR_ SIZE	Number of disk space occupied by the directory
FILESYSTEM	The device file path name corresponding to the file system where the directory is located, usually corresponding to the disk partition
SIZE	The available space capacity of FILESYTEM
USED	Space capacity already occupied
AVAIL	Available Space Capacity
PCT	Percentage of space usage

5.7.4.2.3 CACHE_ USAGE_ INFO

Function Description

Current node database cache usage statistics.

Table Structure Description

Table 5316 Table Structure Information Explanation:

Column Name	Description
HOST	host name
TOTAL_ DC	Total DC
UNLOCKED_ DC	Total number of unlocked DCs
TOTAL_ DC_ SIZE	Total size of DC memory
UNLOCKED_ DC_ SIZE	Total size of unlocked DC memory
HOT_ TOTAL_ DC	Total number of hot DCs
HOT_ UNLOCKED_ DC	Total number of hot DCs unlocked
HOT_ TOTAL_ DC_ SIZE	Total size of hot DC memory
HOT_ UNLOCKED_ DC_ SIZE	Total size of hot DC memory for unlock
COLD_ TOTAL_ DC	Total Cold DC
COLD_ UNLOCKED_ DC	Total number of cold DCs unlocked
COLD_ TOTAL_ DC_ SIZE	Cold DC Total Memory Size
COLD_ UNLOCKED_ DC_ SIZE	Total size of cold DC memory for unlock
HIT_ RATE	DC hit rate

5.7.4.2.4 CACHE_ CELL_ STATUS_ INFO

Function Description

Statistical information of the DC of the database cache.

Table Structure Description

Table 5317 Table Structure Information Explanation:

Column Name	Description
TABLE_ ID	Table ID
COLUMN_ ID	Column ID
DP	DC number
LOCK_ COUNT	Number of locks added
LOCKED	Is it locked
SIZE	Occupy space

5.7.4.2.5 HEAP_ USAGE_ INFO

Function Description

Database memory heap usage statistics.

Table Structure Description

Table 5318 Table Structure Information Explanation:

Column Name	Description
HOST	host name
HEAP_ TYPE	Type of heap: dc_ Heap: Packet cache heap temp_ Heap: temporary memory heap large_ Heap: Large memory cache heap
HEAP_ SIZE	Total heap size
USED_ IN_ HEAP	Heap usage size
USED_ IN_ SYSTEM	The size of additional memory allocated from the system heap when the data heap cannot allocate memory
MAX_ USED_ BLOCK	Peak usage of the current heap
MAX_ FREE_ BLOCK	Peak value of maximum available heap fragments
USED_ BLOCKS	Number of memory requests using the heap (number of allocated memory blocks in the heap)

5.7.4.2.6 SESSION_ MEMORY_ USAGE_ INFO

Function Description

Session level memory statistics.

Table Structure Description

Table -5319 Table Structure Information Explanation:

Column Name	Description
HOST	host name
ID	session id
MEM_ CELL	The size of cache DC data allocated in memory (i.e. heap_data heap)
MEM_ CELL_ SYS	The memory size requested from the system when the DC data cache is insufficient
MEM_ CELL_ PEAK	Peak memory usage in DC data cache
MEM_ LARGE	Heap allocated in memory_ The size of the large heap
MEM_ LARGE_ SYS	The memory size requested from the system when the large heap is insufficient
MEM_ LARGE_ PEAK	Peak memory usage of large heap
MEM_ TEMP	Temporary space allocated in memory heap_ Size of temp heap
MEM_ TEMP_ SYS	The memory size requested from the system when the temp heap is insufficient
MEM_ TEMP_ PEAK	Peak memory usage of temp heap
MEM_ SYS	Memory size used by the operating system
CURRENT	The current memory size used by the session
PEAK	The peak memory used by the session during runtime
PEAK_ TIMESTAMP	The time point at which the session's memory usage reaches its peak
TEMP_ SPACE	Temporary space allocated by session

5.7.4.2.7 MEMORY_ USAGE_ INFO

Function Description

Overview information of the memory section of this node.

Table Structure Description

Table -5320 Table Structure Information Explanation:

Column Name	Description
HOST	Name of this node
PHSICAL_ MEMORY	Physical memory size
SWAP_ SIZE	Swap partition size
PCT	Coefficient for using system memory upper limit for gbased
UPPER_ LIMIT	The upper limit of system memory that gbased can use, calculated by the formula: PHSICAL_ MEMORY* PCT

INIT_ USED	Memory used for initial gbased
CURRENT_ USED	Current memory used by GBased
MEMORY_ PEAK	Peak memory used by GBased during runtime
MEMORY_ PEAK_ TIMESTAMP	Time point when memory usage reaches its peak

5.7.4.2.8 TABLES

Function Description

Statistical information of the table.

Table Structure Description

Table 5321 Table Structure Information Explanation:

Column Name	Description
TABLE_ VC	Virtual cluster name
TABLE_ SCHEMA	Database name
TABLE_ NAME	Table Name
MAX_ ROWID	Rowid max
DELETE_ ROWS	Number of rows deleted
TABLE_ ROWS	Number of data in the table
STORAGE_ SIZE	Size of storage space occupied
DELETABLE_ SIZE	Size of space that can be deleted
SHRINKABLE_ SIZE	The size of space that can be flushed
DELETE_ RATIO	The proportion of deleted data occupied

5.7.4.2.9 MEM_ DETAIL_ CELL

Function Description

CELL heap memory request details.

Table Structure Description

Table 5321 Table Structure Information Explanation:

Column Name	Description
CLASS	Memory marker, range [0511]
SIZE	byte
ADDR	address
INHEAP	In the memory application mechanism of the cell heap, the application is first made in the pre allocated heap

	memory, and when the application cannot be made, an attempt is made to apply from the system. 0: System Memory 1: Memory in cell

5.7.4.2.10 MEM_ DETAIL_ LARGE

Function Description

LARGE heap memory request details.

Table Structure Description

Table -5321 Table Structure Information Explanation:

Column Name	Description
CLASS	Memory marker, range [0511]
SIZE	byte
ADDR	address
INHEAP	In the memory application mechanism of the Large heap, the request is first made in the pre allocated heap memory, and when the request cannot be made, an attempt is made to apply from the system. 0: System Memory 1: Memory in cell

5.7.4.2.11 MEM_ DETAIL_ TEMP

Function Description

TEMP heap memory request details.

Table Structure Description

Table -5321 Table Structure Information Explanation:

Column Name	Description
CLASS	Memory marker, range [0511]
SIZE	byte
ADDR	address
INHEAP	In the memory application mechanism of the TEMP heap, the request is first made in the pre allocated heap memory, and when the request cannot be made, an attempt is made

	to apply from the system. 0: System Memory 1: Memory in cell

5.7.4.2.12 MEM_ DETAIL_ SYS

Function Description

SYS system memory application details.

Table Structure Description

Table -5321 Table Structure Information Explanation:

Column Name	Description
CLASS	Memory marker, range [0511]
SIZE	byte
ADDR	address
INHEAP	In the system memory application mechanism, the application is first made in the pre allocated heap memory, and when the application cannot be made, an attempt is made to apply from the system. 0: System Memory 1: Memory in cell
TYPE	Memory type requested

5.7.4.2.13 CLUSTER_ MONIT_ INFO

Function Description

SYS system memory application details.

Table Structure Description

Table -5321 Table Structure Information Explanation:

Column Name	Description
HOST	Server IP
GCWARE_ STATE	Gcware node status
GCLUSTER_ STATE	Gcluster node status
GNODE_ STATE	Node status of gnode
SYNCSERVER_ STATE	Syncserver service status
COORSERVER_ DATA_ STATUS	GCluster node data consistency status

DATASERVER_ DATA_ STATUS	Node data consistency status of gnode
GCLUSTER_ VMSIZE	
GCLUSTER_ VMRSS	
GCLUSTER_ DISK_ AVAIL_ SIZE	
GCLUSTER_ DISK_ AVAIL_ PERCENT	
GNODE_ VMSIZE	
GNODE_ VMRSS	
GNODE_ DISK_ AVAIL_ SIZE	
GNODE_ DISK_ AVAIL_ PERCENT	
CLUSTER_ STATUS	
CLUSTER_ MODE	

GBASE®

南大通用数据技术股份有限公司
General Data Technology Co., Ltd.

官方微信

GBase 8a 技术社区

技术支持热线：400-013-9696