GaussDB(for MySQL)

FAQs

Issue	01
Date	2024-09-04

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Contents

1 Product Consulting

1.1 What Should I Pay Attention to When Using GaussDB(for MySQL)?

1. Instance operating systems (OSs) are invisible to you. Your applications can access a database only through an IP address and a port.
2. The backup files stored in Object Storage Service (OBS) buckets and the Elastic Cloud Servers (ECSs) used by GaussDB(for MySQL) are invisible to you. They are visible only to the instance management system.
3. When you view your instance in the instance list, select the region where your instance is located.
4. After creating a GaussDB(for MySQL) instance, you do not need to perform basic O&M operations, such as enabling HA and installing security patches. However, you must pay attention to:
   a. vCPUs and memory of your instance. If they become insufficient, you need to change them in a timely manner.
   b. Storage space of your instance. If the storage is used up, you will be billed on a pay-per-use basis for any additional storage, but if you scale up storage in advance, you can pay for the additional storage at yearly/monthly rates.
   c. Performance of your instance. You need to check for slow query SQL statements, SQL statements to be optimized, or redundant or missing indexes regularly.

1.2 What Can I Do About Slow Responses of Websites When They Use GaussDB(for MySQL)?

To solve this problem:

- Check the performance of GaussDB(for MySQL) instances on the GaussDB(for MySQL) console.

- Compare the database connection status of local databases and GaussDB(for MySQL) instances. This problem depends on web applications.

2 Resource Freezing, Unfreezing, Release, Deletion, and Unsubscription

Why Are My Resources Released?

If your subscriptions have expired but not been renewed, or you are in arrears due to insufficient balance, your resources enter a grace period. If you do not renew the subscriptions or top up your account after the grace period expires, your resources will enter a retention period. During the retention period, the resources are not available. If the renewal is still not completed or the outstanding amount is still not paid off when the retention period ends, the stored data will be deleted and the cloud service resources will be released. For details, see **Service Suspension and Resource Release**.

Why Are My Resources Frozen?

Your resources may be frozen for a variety of reasons. The most common reason is that you are in arrears.

Can I Still Back Up Data If My DB Instance Is Frozen?

No. If your instance is frozen due to arrears, you need to unfreeze the instance first.

How Do I Unfreeze My Resources?

Frozen due to arrears: You can renew your resources or top up your account. Instances frozen due to arrears can be renewed, released, or deleted. Yearly/Monthly instances that have expired cannot be unsubscribed from, while those that have not expired can be unsubscribed from.

What Happens When My Resources Are Frozen, Unfrozen, or Released?

- After your resources are frozen:
  - They cannot be accessed, causing downtime. For example, if your instance is frozen, it cannot be connected.
  - If they are yearly/monthly resources, no changes can be made to them.

    - They can be unsubscribed from or deleted manually.
- After your resources are unfrozen, you can connect to them again.
- If your resources are released, your instance will be deleted.

How Do I Renew My Resources?

After a yearly/monthly instance expires, you can renew it on the **Renewals** page. For details, see **Renewal Management**.

Can My Resources Be Recovered After Being Released? /Can I Retrieve an Incorrect Unsubscription?

You can restore your deleted instance from a manual backup or rebuild your instance in the recycle bin during the retention period. For details, see **Restoring Data to a DB Instance** and **Rebuilding a Deleted Instance from Recycle Bin**.

Before unsubscribing from a resource, confirm the resource information carefully. If you have unsubscribed from an instance by mistake, purchase a new one.

How Do I Delete My Instance?

- For pay-per-use instances, see **Deleting a DB Instance Billed on a Pay-per-Use Basis**.
- For yearly/monthly instances, see **Unsubscribing from a Yearly/Monthly Instance**.

3 Database Connections

3.1 What Should I Do If I Can't Connect to My GaussDB(for MySQL) Instance?

Possible Causes

Try the following:

1. **Check whether the DB instance is available.**

   for example, the DB instance status is abnormal.

2. **(Common) Check whether the client connection is correct.**
   - If you connect to a DB instance over a private network, ensure that the DB instance and ECS are in the same region and VPC.
   - If you connect to a DB instance over a public network, bind an EIP to the DB instance and then connect to the DB instance through the EIP.

3. **Check the SSL connection is used.**

   Run either of the following example commands to enable or disable SSL:
   - SSL enabled: **mysql -h 172.16.0.31 -P 3306 -u root -p --ssl-ca=/tmp/ca.pem**
   - SSL disabled: **mysql -h 172.16.0.31 -P 3306 -u root -p**

4. **Check whether the parameters in the connection command are correct.**

   For example, check whether the following parameters are configured correctly: connection address, port number, username, password, SSL connection parameters, and JDBC connection parameters.

5. **(Common) Check whether the network connection is normal.**
   - For a private network connection:
     i. Check whether the ECS and DB instance are in the same region and VPC.
     ii. Check security group rules.
     iii. On the ECS, check whether the DB instance port can be connected to.

– For a public network connection:

   i. Check security group rules.

   ii. Check network ACL rules.

   iii. Ping the affected EIP from another ECS in the same region.

6. **(Common) Check whether there are too many connections to the DB instance.**

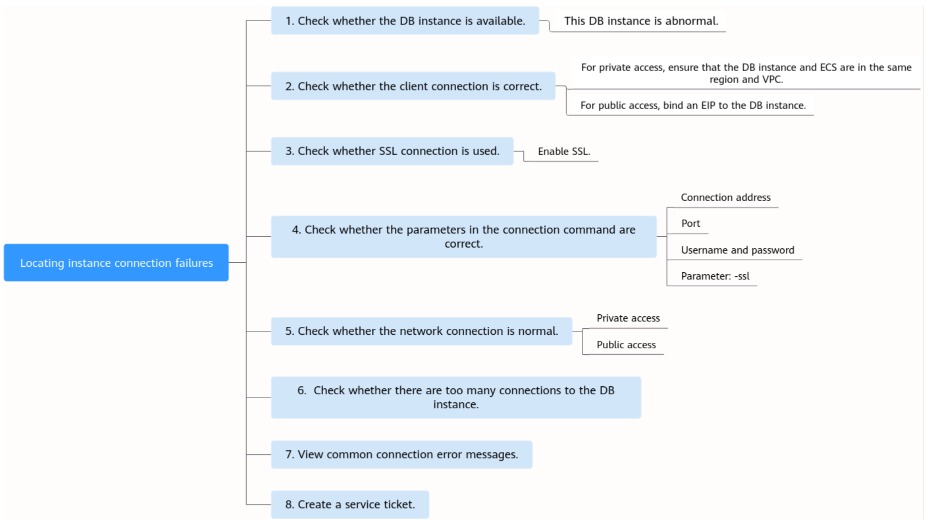
   If there is an excessive number of database connections, applications may be unable to connect.

7. **View common connection error messages.**

   Find corresponding solutions based on connection error messages.

Fault Locating

Figure 3-1 Locating instance connection failures


Locating instance connection failures
1. Check whether the DB instance is available.
This DB instance is abnormal.
2. Check whether the client connection is correct.
For private access, ensure that the DB instance and ECS are in the same region and VPC.
For public access, bind an EIP to the DB instance.
3. Check whether SSL connection is used.
Enable SSL.
4. Check whether the parameters in the connection command are correct.
Connection address
Port
Username and password
Parameter: -ssl
5. Check whether the network connection is normal.
Private access
Public access
6. Check whether there are too many connections to the DB instance.
7. View common connection error messages.
8. Create a service ticket.

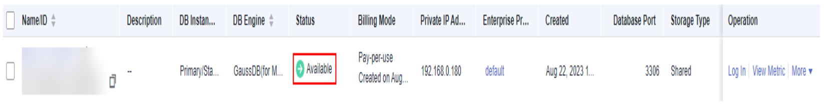
1. **Check whether the DB instance is available.**

   Check whether the DB instance is in the **Available** state.

   Possible cause: The DB instance is abnormal.

   Solution: If the DB instance is abnormal, reboot it.

   Figure 3-2 Checking DB instance status



Name/ID	Description	DB Instan...	DB Engine	Status	Billing Mode	Private IP Ad...	Enterprise Pr...	Created	Database Port	Storage Type	Operation
[redacted]	--	Primary/Sta...	GaussDB(for M...	Available	Pay-per-use Created on Aug...	192.168.0.180	default	Aug 22, 2023 1...	3306	Shared	Log In | View Metric | More ▾

2. **Check whether the client connection is correct.**

   Install a **MySQL 8.0 client**.

   For details about how to connect to a DB instance over a private or public network, see **Can an External Server Access a GaussDB(for MySQL) Instance?**

Table 3-1 Connection model

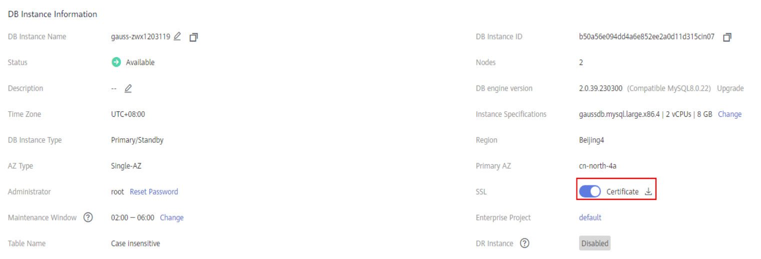
Connection method	Scenario	Example
Private network	A private IP address is provided by default. If your applications are deployed on an ECS that is in the same region and VPC as the DB instance, connect to the ECS and DB instance through a private IP address.	**mysql -h** ***private_IP_address*** **-P 3306 -u root -p --ssl-ca=/tmp/ca.pem** Go to the **Basic Information** page of the DB instance and view the private IP address in the **Network Information** area.
Public network	If you cannot access a DB instance through a private IP address, bind an EIP to the DB instance and connect the instance to the ECS (or a cloud server in a public network) through the EIP. For EIP pricing details, see **EIP billing details**.	**mysql -h** ***EIP*** **-P 3306 -u root -p --ssl-ca=/tmp/ca.pem** Go to the **Basic Information** page of the DB instance and view the EIP in the **Network Information** area.

3. **Check whether the SSL connection is used.**
   - (Recommended) Enable SSL on the **Basic Information** page of the instance, download and decompress the package, and upload the root certificate **ca.pem** to the **/tmp** directory of the ECS.

     Example:

     mysql -h 172.16.0.31 -P 3306 -u root -p --ssl-ca=/tmp/ca.pem

     Figure 3-3 Enabling SSL

     DB Instance Information

DB Instance Name	gauss-zwx1203119	DB Instance ID	b50a56e094dd4a6e852ee2a0d11d315cin07
Status	Available	Nodes	2
Description	--	DB engine version	2.0.39.230300 (Compatible MySQL8.0.22) Upgrade
Time Zone	UTC+08:00	Instance Specifications	gaussdb.mysql.large.x86.4 | 2 vCPUs | 8 GB Change
DB Instance Type	Primary/Standby	Region	Beijing4
AZ Type	Single-AZ	Primary AZ	cn-north-4a
Administrator	root Reset Password	SSL	Certificate
Maintenance Window	02:00 – 06:00 Change	Enterprise Project	default
Table Name	Case insensitive	DR Instance	Disabled

   - Common connection: Disable SSL on the **Basic Information** page.

     Example:

     mysql -h 172.16.0.31 -P 3306 -u root -p
4. **Check whether the parameters in the connection command are correct.**

   Ensure that the connection address, port, username and password, and SSL certificate are correct, and try to connect to the DB instance again.
   - **Connecting to a DB instance over a private network**

- Connection command

  mysql -h *connection_address* **-P** *database_port* **-u** *username* **-p --ssl-ca=** *SSL_certificate_name*

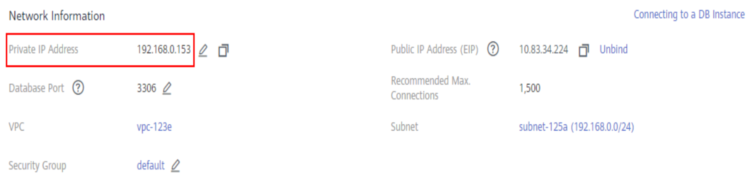
  Example:

  mysql -h 192.168.0.153 -P 3306 -u root -p --ssl-ca=/tmp/ca.pem

- *connection_address*

  Go to the **Basic Information** page of the DB instance and view the private IP address in the **Network Information** area.

  Figure 3-4 Private IP address



Network Information | Connecting to a DB Instance

Private IP Address	192.168.0.153	Public IP Address (EIP)	10.83.34.224 Unbind
Database Port	3306	Recommended Max. Connections	1,500
VPC	vpc-123e	Subnet	subnet-125a (192.168.0.0/24)
Security Group	default		

- *database_port*

  Go to the **Basic Information** page of the DB instance and view the database port in the **Network Information** area.

- *username*

  Enter **root** and its password.

  Check whether the username or password is correct if an error similar to the following occurs during the database connection:

```
[Warning] Access denied for user 'username'@'yourIp' (using password: NO)
[Warning] Access denied for user 'username'@'yourIp' (using password: YES)
```

- *SSL_certificate_name*

  Enter the name of the SSL certificate file. The path and file name must be the same as those in the command.

– **Connecting to a DB instance over a public network**

- Connection command

  mysql -h *connection_address* **-P** *database_port* **-u** *username* **-p --ssl-ca=** *SSL_certificate_name*

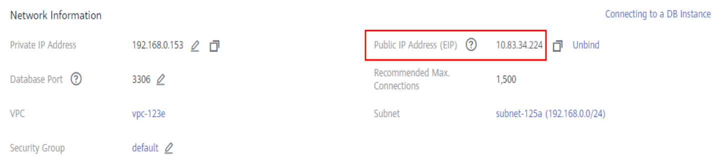
  Example:

  mysql -h 10.83.34.224 -P 3306 -u root -p --ssl-ca=/tmp/ca.pem

- *connection_address*

  Go to the **Basic Information** page of the DB instance and view the EIP in the **Network Information** area.

  Figure 3-5 EIP



Network Information | Connecting to a DB Instance

Private IP Address	192.168.0.153	Public IP Address (EIP)	10.83.34.224 Unbind
Database Port	3306	Recommended Max. Connections	1,500
VPC	vpc-123e	Subnet	subnet-125a (192.168.0.0/24)
Security Group	default		

- *database_port*

  Go to the **Basic Information** page of the DB instance and view the database port in the **Network Information** area.

- *username*

  Enter **root** and its password.

  ```
  [Warning] Access denied for user 'username'@'yourIp' (using password: NO)
  [Warning] Access denied for user 'username'@'yourIp' (using password: YES)
  ```

- *SSL_certificate_name*

  Enter the name of the SSL certificate file. The path and file name must be the same as those in the command.

5. **Check whether the network connection is normal.**

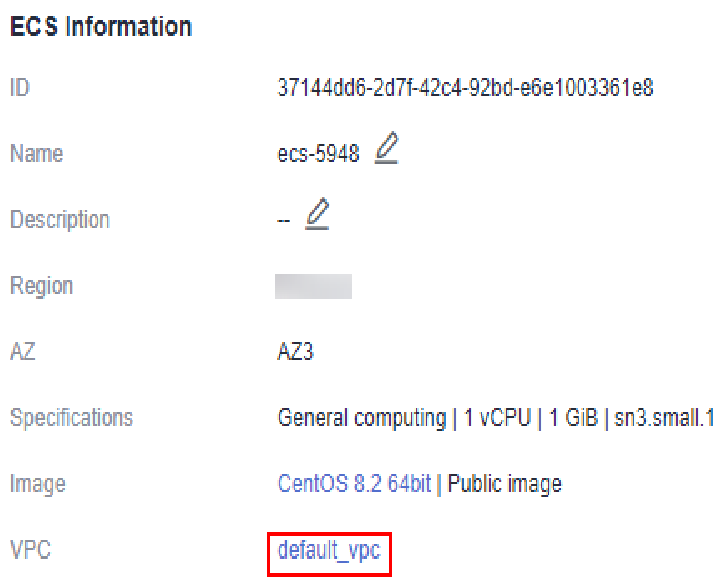
   For a private network connection:

   a. Check whether the ECS and DB instance are in the same region and VPC.

      - If the ECS and DB instance are in different regions, they cannot communicate with each other. Select a region near to your service area to reduce network latency and experience faster access. To connect to the DB instance across regions, use Cloud Connect (CC) or Virtual Private Network (VPN). For details, see **Can I Access a GaussDB(for MySQL) Instance over an Intranet Connection Across Regions?**

      - If the ECS and DB instance are in different VPCs of the same region, they cannot communicate with each other through a private network. After a DB instance is created, you cannot change its VPC. In this case, create a VPC peering connection. For details, see **What Should I Do If the Network Connectivity Test Fails?**

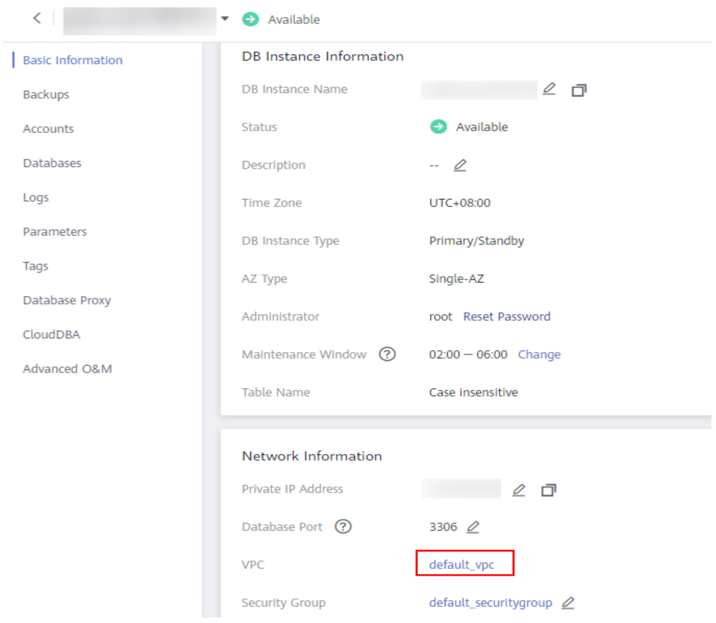
Figure 3-6 Checking the VPC of an ECS



ECS Information

ID	37144dd6-2d7f-42c4-92bd-e6e1003361e8
Name	ecs-5948
Description	--
Region	
AZ	AZ3
Specifications	General computing | 1 vCPU | 1 GiB | sn3.small.1
Image	CentOS 8.2 64bit | Public image
VPC	default_vpc

Figure 3-7 Checking the VPC of a DB instance


[redacted] Available

- Basic Information
- Backups
- Accounts
- Databases
- Logs
- Parameters
- Tags
- Database Proxy
- CloudDBA
- Advanced O&M

DB Instance Information

DB Instance Name	[redacted]
Status	Available
Description	--
Time Zone	UTC+08:00
DB Instance Type	Primary/Standby
AZ Type	Single-AZ
Administrator	root Reset Password
Maintenance Window	02:00 — 06:00 Change
Table Name	Case insensitive

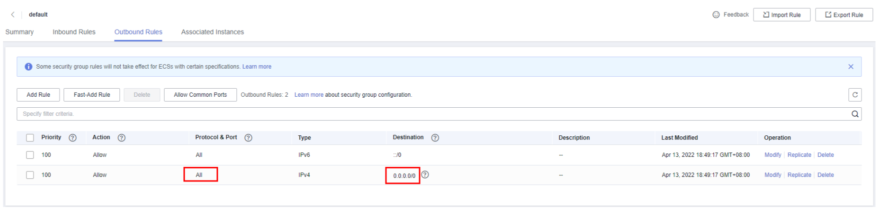
Network Information

Private IP Address	[redacted]
Database Port	3306
VPC	default_vpc
Security Group	default_securitygroup

b. Check security group rules.

- If **Destination** is not **0.0.0.0/0** and **Protocol & Port** is not **All** on the **Outbound Rules** page of the ECS, add the private IP address and port of the DB instance to the outbound rules.

  Figure 3-8 ECS security group

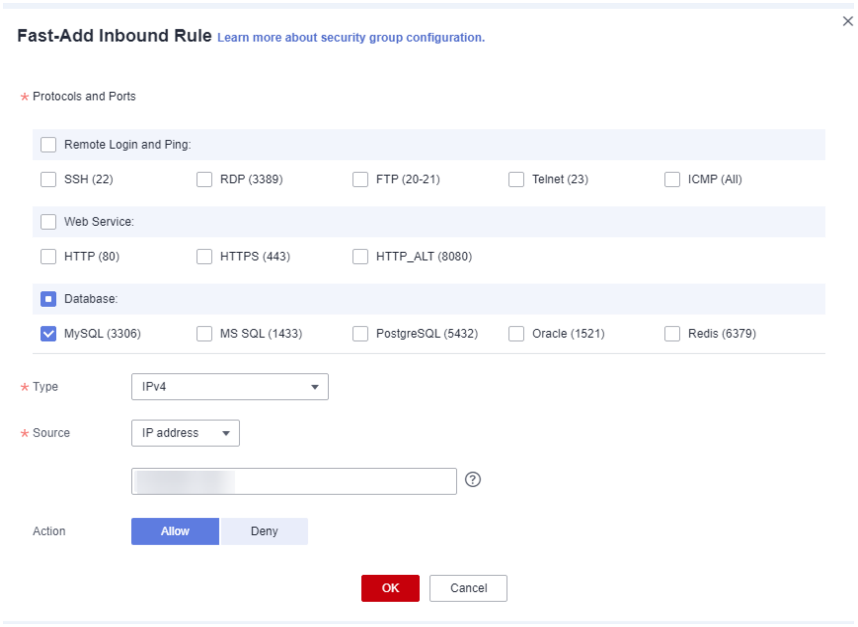


Some security group rules will not take effect for ECSs with certain specifications. Learn more

Outbound Rules: 2 Learn more about security group configuration.

Priority	Action	Protocol & Port	Type	Destination	Description	Last Modified	Operation
100	Allow	All	IPv6	::/0	--	Apr 13, 2022 18:49:17 GMT+08:00	Modify | Replicate | Delete
100	Allow	All	IPv4	0.0.0.0/0	--	Apr 13, 2022 18:49:17 GMT+08:00	Modify | Replicate | Delete

Figure 3-9 Fast adding an outbound rule (private IP address of a DB instance)



Fast-Add Inbound Rule Learn more about security group configuration.

* Protocols and Ports

☐ Remote Login and Ping:				
☐ SSH (22)	☐ RDP (3389)	☐ FTP (20-21)	☐ Telnet (23)	☐ ICMP (All)
☐ Web Service:				
☐ HTTP (80)	☐ HTTPS (443)	☐ HTTP_ALT (8080)		
■ Database:				
✓ MySQL (3306)	☐ MS SQL (1433)	☐ PostgreSQL (5432)	☐ Oracle (1521)	☐ Redis (6379)

* Type: IPv4

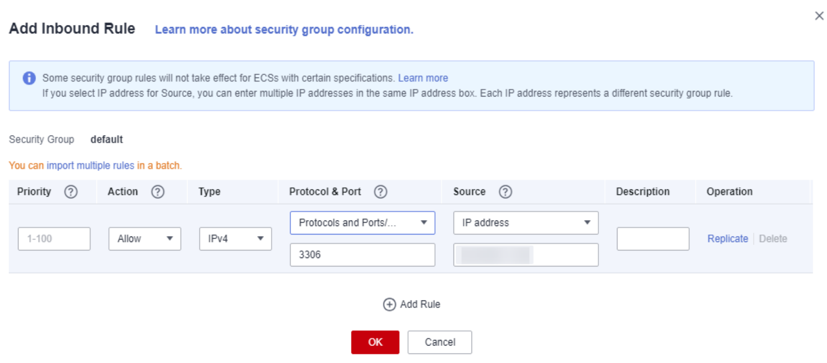
* Source: IP address

[redacted]

Action: [✓ Allow] Deny

OK Cancel

Figure 3-10 Adding an outbound rule (private IP address of a DB instance)



Add Inbound Rule Learn more about security group configuration.

Some security group rules will not take effect for ECSs with certain specifications. Learn more
If you select IP address for Source, you can enter multiple IP addresses in the same IP address box. Each IP address represents a different security group rule.

Security Group default

You can import multiple rules in a batch.

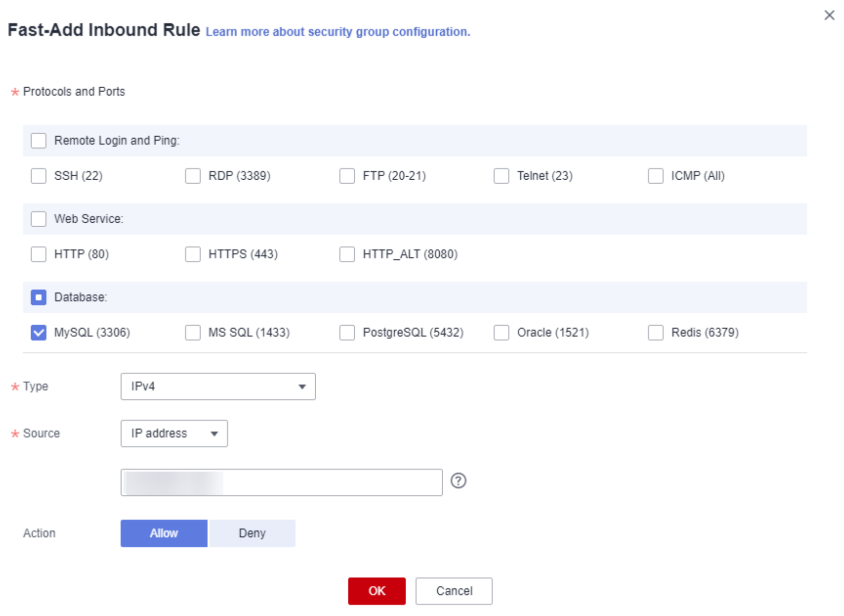
Priority	Action	Type	Protocol & Port	Source	Description	Operation
1-100	Allow	IPv4	Protocols and Ports/... 3306	IP address [redacted]		Replicate | Delete

Add Rule

OK Cancel

- Add the private IP address and port of the ECS to the **inbound rules** of the DB instance.

Figure 3-11 Fast adding an inbound rule (private IP address of an ECS)



Fast-Add Inbound Rule Learn more about security group configuration.

* Protocols and Ports

- [ ] Remote Login and Ping:
  - [ ] SSH (22) - [ ] RDP (3389) - [ ] FTP (20-21) - [ ] Telnet (23) - [ ] ICMP (All)
- [ ] Web Service:
  - [ ] HTTP (80) - [ ] HTTPS (443) - [ ] HTTP_ALT (8080)
- [■] Database:
  - [✓] MySQL (3306) - [ ] MS SQL (1433) - [ ] PostgreSQL (5432) - [ ] Oracle (1521) - [ ] Redis (6379)

* Type: IPv4

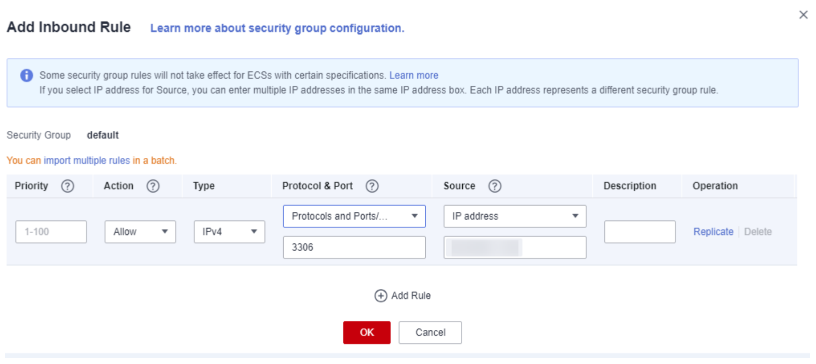
* Source: IP address

[redacted]

Action: [✓ Allow] Deny

OK Cancel

Figure 3-12 Adding an inbound rule (private IP address of an ECS)



Add Inbound Rule Learn more about security group configuration.

Some security group rules will not take effect for ECSs with certain specifications. Learn more
If you select IP address for Source, you can enter multiple IP addresses in the same IP address box. Each IP address represents a different security group rule.

Security Group default

You can import multiple rules in a batch.

Priority	Action	Type	Protocol & Port	Source	Description	Operation
1-100	Allow	IPv4	Protocols and Ports/... 3306	IP address [redacted]		Replicate Delete

Add Rule

OK Cancel

c. On the ECS, check whether the private IP address of the DB instance can be connected to.

telnet *private_IP_address port*

Example:

telnet 192.168.0.153 3306

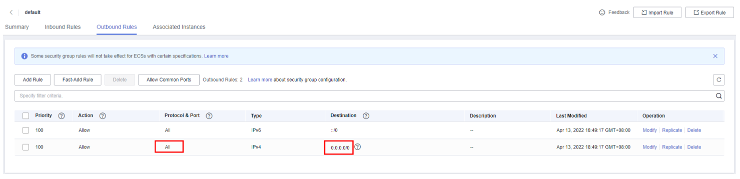
- If the connection is normal, the network is normal.
- If the connection fails, **create a service ticket** to contact customer service for assistance.

For a public network connection:

a. Check security group rules.

- If **Destination** is not **0.0.0.0/0** and **Protocol & Port** is not **All** on the **Outbound Rules** page of the ECS, add the EIP and port of the DB instance to the outbound rules.

Figure 3-13 ECS security group



default

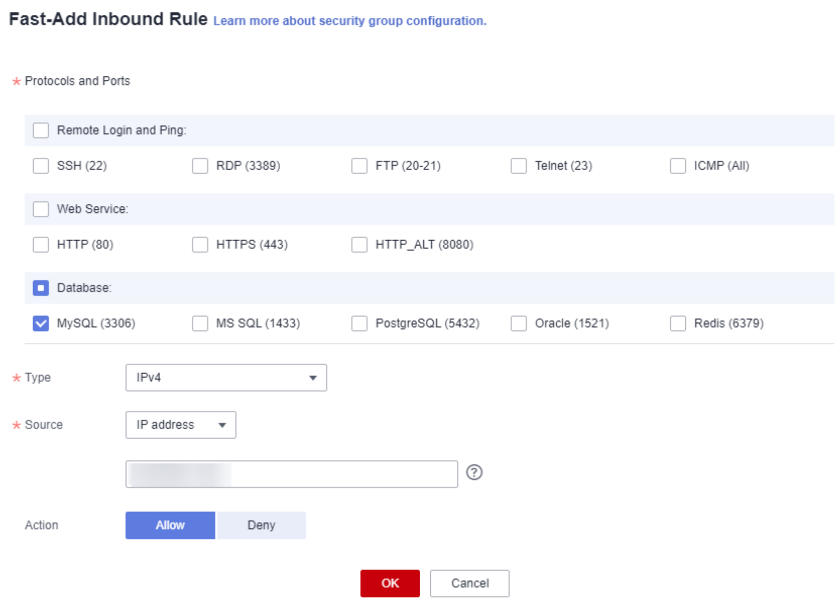
Summary | Inbound Rules | Outbound Rules | Associated Instances

Some security group rules will not take effect for ECSs with certain specifications. Learn more

Outbound Rules: 2 Learn more about security group configuration.

Priority	Action	Protocol & Port	Type	Destination	Description	Last Modified	Operation
100	Allow	All	IPv6	::/0	--	Apr 13, 2022 18:49:17 GMT+08:00	Modify | Replicate | Delete
100	Allow	All	IPv4	0.0.0.0/0	--	Apr 13, 2022 18:49:17 GMT+08:00	Modify | Replicate | Delete

Figure 3-14 Fast adding an outbound rule (DB instance EIP)



Fast-Add Inbound Rule Learn more about security group configuration.

* Protocols and Ports

Remote Login and Ping: SSH (22), RDP (3389), FTP (20-21), Telnet (23), ICMP (All)

Web Service: HTTP (80), HTTPS (443), HTTP_ALT (8080)

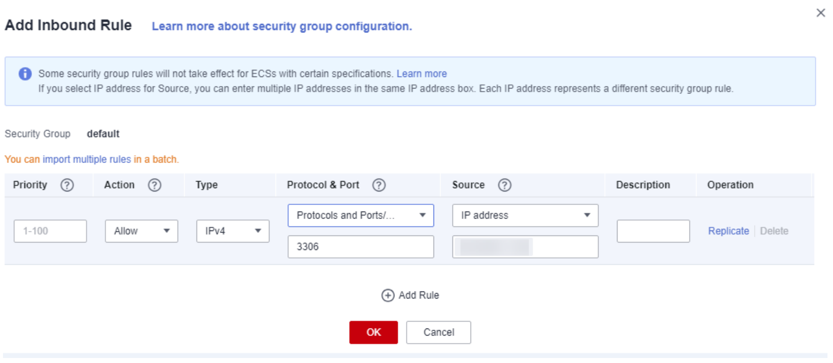
Database: [✓ MySQL (3306)], MS SQL (1433), PostgreSQL (5432), Oracle (1521), Redis (6379)

* Type: IPv4

* Source: IP address

Action: [✓ Allow] Deny

Figure 3-15 Adding an outbound rule (DB instance EIP)



Add Inbound Rule Learn more about security group configuration.

Some security group rules will not take effect for ECSs with certain specifications. Learn more
If you select IP address for Source, you can enter multiple IP addresses in the same IP address box. Each IP address represents a different security group rule.

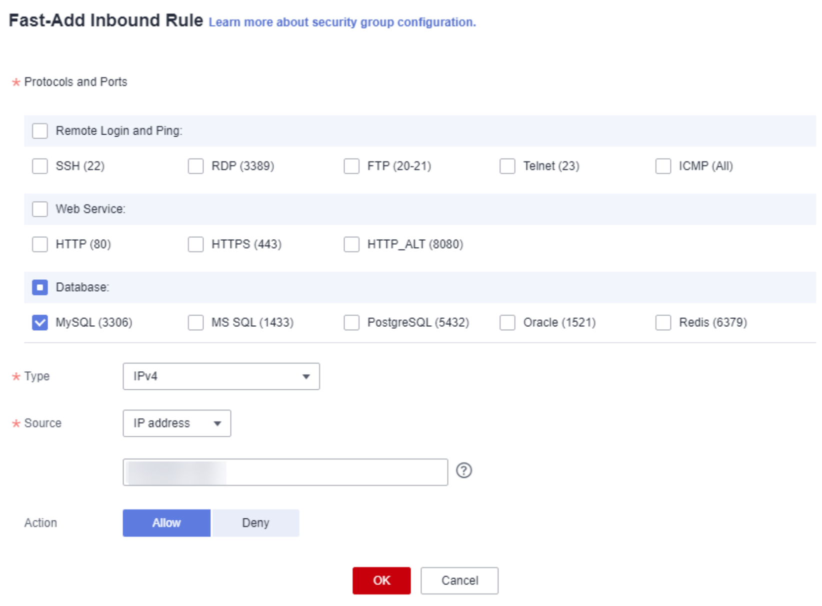
Security Group default

You can import multiple rules in a batch.

Priority	Action	Type	Protocol & Port	Source	Description	Operation
1-100	Allow	IPv4	Protocols and Ports/... 3306	IP address		Replicate | Delete

- Add the EIP and port of the ECS to the **inbound rules** of the DB instance.

Figure 3-16 Adding an inbound rule (ECS EIP)


Fast-Add Inbound Rule Learn more about security group configuration.

* Protocols and Ports

☐ Remote Login and Ping:				
☐ SSH (22)	☐ RDP (3389)	☐ FTP (20-21)	☐ Telnet (23)	☐ ICMP (All)
☐ Web Service:				
☐ HTTP (80)	☐ HTTPS (443)	☐ HTTP_ALT (8080)		
■ Database:				
☑ MySQL (3306)	☐ MS SQL (1433)	☐ PostgreSQL (5432)	☐ Oracle (1521)	☐ Redis (6379)

* Type: IPv4

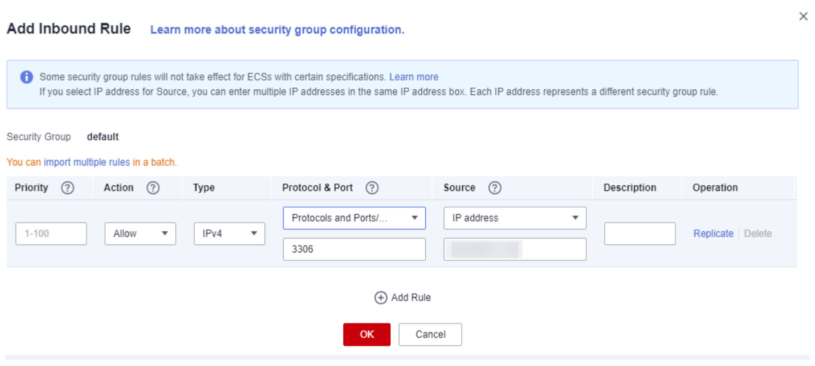
* Source: IP address

[redacted]

Action: [✓ Allow] Deny

OK Cancel

Figure 3-17 Adding an inbound rule (ECS EIP)


Add Inbound Rule Learn more about security group configuration.

Some security group rules will not take effect for ECSs with certain specifications. Learn more
If you select IP address for Source, you can enter multiple IP addresses in the same IP address box. Each IP address represents a different security group rule.

Security Group **default**

You can import multiple rules in a batch.

Priority	Action	Type	Protocol & Port	Source	Description	Operation
1-100	Allow	IPv4	Protocols and Ports/... 3306	IP address [redacted]		Replicate | Delete

Add Rule

OK Cancel

b. Check network ACL rules.

   i. Go to the **Network ACLs** page.

   ii. Check whether the NIC to which the DB instance and EIP are bound belongs to the subnet associated with the network ACL.

   iii. Check whether the network ACL is enabled.

   If it is, **add an ICMP rule to allow traffic**.

   The default network ACL rule denies all inbound and outbound packets. This default rule is still applied even if the network ACL is disabled.

c. Ping the affected EIP from another ECS in the same region.

   If you cannot ping the DB instance's EIP from an ECS, try pinging it from another ECS in the same region. If the EIP can be pinged, the network is normal. In this case, **create a service ticket** to contact customer service.

6. **Check whether there are too many connections to the DB instance.**

   Check method:

a. **Log in to the management console.**

b. Click in the upper left corner and select a region and project.

c. Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.

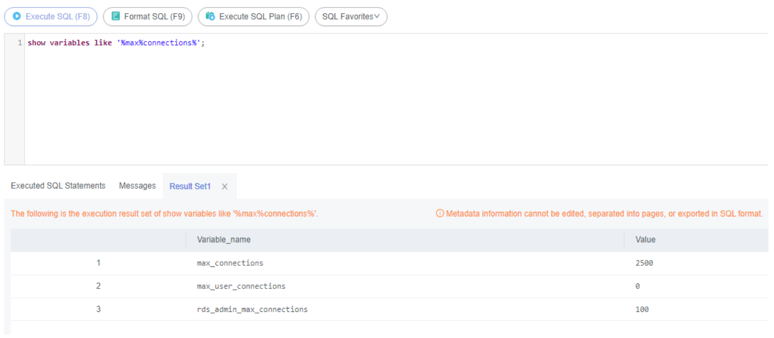
d. On the **Instances** page, locate the DB instance and click **Log In** in the **Operation** column.

e. Enter the password and click **Test Connection**. After the connection is successful, click **Log In** to access DAS.

f. Choose **SQL Operations** > **SQL Query**.

g. Enter the command and click **Execute SQL** to check the number of instance connections.

show variables like '%max%connections%';



```
show variables like '%max%connections%';
```

The following is the execution result set of show variables like '%max%connections%'.

Metadata information cannot be edited, separated into pages, or exported in SQL format.

	Variable_name	Value
1	max_connections	2500
2	max_user_connections	0
3	rds_admin_max_connections	100

- **max_connections**: the maximum number of clients that can be connected at the same time. If this parameter is set to **default**, the maximum number of clients depends on the amount of memory configured. For details, see **What Is the Maximum Number of Connections to a GaussDB(for MySQL) Instance?**.

- **max_user_connections**: the maximum number of concurrent connections allowed for a specific GaussDB(for MySQL) account.

h. Check whether the total connections and current active connections have reached the upper limits by referring to **Viewing Instance Monitoring Metrics**. Determine whether to release the connections.

Possible cause: If there are too many database connections, applications may be unable to connect, and full and incremental backups may fail, affecting services.

Solution:

a. Check whether applications are connected, optimize the connections, and release unnecessary connections.

b. If this parameter is set to **default**, you can scale up the DB instance to set **max_connections** to a larger value. For details, see **Changing vCPUs and Memory of a DB Instance**.

c. Check whether any metrics are abnormal and whether any alarms are generated on the Cloud Eye console. Cloud Eye monitors database metrics, such as the CPU usage, memory usage, storage space usage, and database connections, and allows you to configure alarm policies to identify risks in advance if any alarms are generated. For details about the supported monitoring metrics, see **Introducing GaussDB(for MySQL) Metrics**.

7. **View common connection error messages.**

   When you run commands to connect to a DB instance, understanding the error messages can help:

   – ERROR 2013: Lost connection to MySQL server during query

     If the values of **wait_timeout** and **interactive_timeout** are too small, the GaussDB(for MySQL) client will automatically disconnect the timeout empty connection. For details, see **Client Automatically Disconnected from a DB Instance**.

   – ERROR 1045 : Access denied for user 'root'@'192.168.0.30' (using password: YES)

     Check whether the password is correct and whether the ECS has the permission to connect to the DB instance. For details, see **"Access denied" Displayed During Database Connection**.

   – Error message "SSL routines: tls_early_post_process_client_hello:unsupported protocol"

     Check the TLS version of GaussDB(for MySQL), and upgrade the TLS version of the client. For details, see **SSL Connection Failed Due to Inconsistent TLS Versions**.

   – Error reported when JDBC is used to connect to the database: "unable to find certification path to requested target"

     The JAR package of MariaDB is used to connect to the database, which is slightly different from the official driver package of MySQL. For details, see **Failed to Connect to a Database Using mariadb-connector in SSL Mode**.

8. If the problem persists, **create a service ticket**.

3.2 What Should I Do If an ECS Can't Connect to a GaussDB(for MySQL) Instance?

Perform the following steps to identify the problem:

Step 1 Check whether the ECS and GaussDB(for MySQL) instance are in the same VPC.

- If they are in the same VPC, go to **Step 2**.
- If they are in different VPCs, create an ECS in the VPC where the GaussDB(for MySQL) instance is located.

Step 2 Check whether a security group has been created for the ECS.

- If a security group has been created, check whether its rules are appropriate.

  For details, see "Connecting to a DB Instance" > "Configuring Security Group Rules" in *GaussDB(for MySQL) User Guide*. Then, go to **Step 3**.
- If no security group has been created, go to the VPC console from the ECS details page and create a security group.

Step 3 On the ECS, check whether the GaussDB(for MySQL) instance port can be connected to.

The default port of a GaussDB(for MySQL) primary/standby instance is **3306**.

```
telnet <connection_address> {port}
```

- If the ECS can connect to the instance, no further action is required.
- If the ECS still cannot connect to the instance port, contact technical support.

----End

3.3 Can an External Server Access a GaussDB(for MySQL) Instance?

An Instance Bound with an EIP

For a GaussDB(for MySQL) instance that has been bound with an EIP, you can access it through the EIP.

For details, see:

Connecting to a DB Instance over a Public Network

An Instance Not Bound with an EIP

- Enable a VPN in a VPC and use the VPN to connect to a GaussDB(for MySQL) instance.
- Create a GaussDB(for MySQL) instance and an ECS in the same VPC and access the GaussDB(for MySQL) instance through the ECS.

For details, see:

Connecting to a DB Instance over a Private Network

3.4 What Is the Maximum Number of Connections to a GaussDB(for MySQL) Instance?

GaussDB(for MySQL) does not have constraints on the number of connections. This number is determined by the default value and value range of the DB engine. For example, you can set **max_connections** and **max_user_connections** in a parameter template to configure the maximum number of connections for a GaussDB(for MySQL) instance.

Changing the Maximum Number of Connections

The number of connections can be changed online. For details, see **Modifying a Parameter Template**.

You can run commands to change the maximum number of connections.

1. Check the maximum number of connections:

   show global variables like 'max_connections';

2. Change the value of **max_connections** under **mysqld** in the **my.cnf** file.

   [mysqld]

max_connections = 1000

About max_connections

max_connections indicates the maximum number of clients that can be connected at the same time. If this parameter is set to **default**, it is related to the instance memory (unit: GB). The calculation formula is as follows:

Estimated value of max_connections = Available node memory/Estimated memory occupied by a single connection

- Available node memory = Total memory – Memory occupied by the buffer pool – 1 GB (mysqld process, OS, and monitoring program)
- Estimated memory usage of a single connection (single_thread_memory) = thread_stack (256 KB) + binlog_cache_size (32 KB) + join_buffer_size (256 KB) + sort_buffer_size (256 KB) + read_buffer_size (128 KB) + read_rnd_buffer_size (256 KB) ≈ 1 MB

The following table lists the default values of **max_connections** for different memory specifications.

Table 3-2 max_connections for different memory specifications

Memory (GB)	Connections
512	100,000
384	80,000
256	60,000
128	30,000
64	18,000
32	10,000
16	5,000
8	2,500
4	1,500
2	800

3.5 What Do I Do If the Number of GaussDB(for MySQL) Database Connections Reaches the Upper Limit?

The number of database connections indicates the number of applications that can be simultaneously connected to a database, and is irrelevant to the maximum number of users allowed by your applications or websites.

If there is an excessive number of database connections, applications may fail to be connected, and the full and incremental backups may fail, affecting services.

Fault Locating

1. Check whether applications are connected, optimize the connections, and release unnecessary connections.
2. Check the specifications and scale them up if needed.
3. Check whether any metrics are abnormal and whether any alarms are generated on the Cloud Eye console. Cloud Eye monitors database metrics, such as the CPU usage, memory usage, storage space usage, and database connections, and allows you to configure alarm policies to identify risks in advance if any alarms are generated. For details, see the *Cloud Eye User Guide*.

Solution

1. Connect to an instance through a private network. Using a private network prevents congestion caused by insufficient bandwidth.

   For details, see:

   Connecting to a DB Instance over a Private Network
2. On the console, set the parameter **innodb_adaptive_hash_index** to **off** to reduce lock wait time.

   For details, see **Modifying a Parameter Template**.
3. Optimize slow queries.

3.6 Are There Any Potential Risks If There Are Too Many Connections to a GaussDB(for MySQL) Instance?

If there is an excessive number of GaussDB(for MySQL) connections, applications may fail to be connected, and the full and incremental backups may fail, affecting services.

Solution

1. Check whether applications are connected, optimize the connections, and release unnecessary connections.
2. Cloud Eye monitors database metrics, such as the CPU usage, memory usage, storage space usage, and database connections, and allows you to set alarm policies to identify potential risks if any alarms are generated.

3.7 What Should I Do If the Network Connectivity Test Fails?

Before connecting to a GaussDB(for MySQL) instance, you need to test the network connectivity to ensure that the client can communicate with the GaussDB(for MySQL) instance.

This section analyzes common causes of network connection failures and provides solutions.

Fault Locating

1. Check whether the ECS and GaussDB(for MySQL) instance are in the same VPC.
2. Check the security group rules, ACL rules, internal network configurations, and ports of the ECS.
3. Check the security group rules of the GaussDB(for MySQL) instance.
4. Ping the affected IP address from another ECS in the same region.

Solution

1. Check whether the ECS and GaussDB(for MySQL) instance are in the same VPC.
   - If they are in the same VPC, go to **2**.
   - If they are in different VPCs of a region, they cannot communicate with each other through a private network.

     To resolve this issue, do as follows:

     i. Create a VPC peering connection. For details, see **VPC Peering Connection Overview**.

     ii. Change the VPC of the ECS to that of the GaussDB(for MySQL) instance. For details, see **Changing a VPC**.
2. Check the ECS configurations.

   For details, see **Why Does Communication Fail Between Two ECSs in the Same VPC or Packet Loss Occur When They Communicate?**
3. Check the security group rules of the GaussDB(for MySQL) instance.

   On the **Inbound Rules** page of the GaussDB(for MySQL) instance security group, add an inbound rule for the private IP address and port of the ECS.
4. Ping the affected IP address from another ECS in the same region.

   On the ECS, ping the IP address that failed to be pinged. If the IP address can be pinged, the virtual network is normal.
5. If the problem persists, submit a service ticket by choosing **Service Tickets > Create Service Ticket** in the upper right corner of the management console.

3.8 Can I Access a GaussDB(for MySQL) Instance over an Intranet Connection Across Regions?

By default, DB instances cannot be accessed over an intranet across regions. Cloud services in different regions cannot communicate with each other over an intranet. You can use Cloud Connect (CC) or Virtual Private Network (VPN) to connect to instances across regions.

- CC allows you to connect two VPCs of the same account or different accounts even if they are in different regions. For details, see **Communication Among VPCs of the Same Account**.

- VPN uses an encrypted tunnel to connect VPCs in different regions and sends traffic over the Internet. It is inexpensive, easy to configure, and easy to use. However, the quality of VPN connections depends on the quality of Internet connections. For details, see **Connecting an On-Premises Data Center to a VPC Through a VPN**.

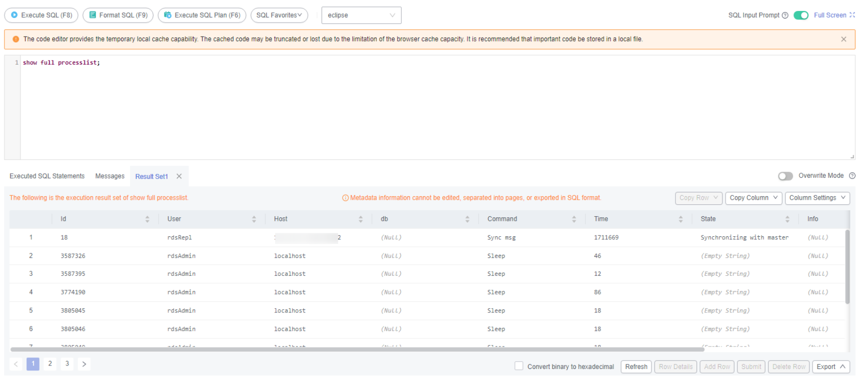
3.9 How Do I Check the Connections to a GaussDB(for MySQL) Instance?

Use either of the following methods:

- Log in to the instance as user **root** and run the following command to view the threads running on it:

  show full processlist;

  Figure 3-18 Viewing threads

  The code editor provides the temporary local cache capability. The cached code may be truncated or lost due to the limitation of the browser cache capacity. It is recommended that important code be stored in a local file.

  ```
  show full processlist;
  ```

  The following is the execution result set of show full processlist.

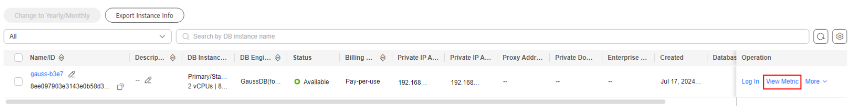
  Metadata information cannot be edited, separated into pages, or exported in SQL format.

	Id	User	Host	db	Command	Time	State	Info
1	18	rdsRepl	[redacted]2	(Null)	Sync msg	1711669	Synchronizing with master	(Null)
2	3587326	rdsAdmin	localhost	(Null)	Sleep	46	(Empty String)	(Null)
3	3587395	rdsAdmin	localhost	(Null)	Sleep	12	(Empty String)	(Null)
4	3774190	rdsAdmin	localhost	(Null)	Sleep	86	(Empty String)	(Null)
5	3805045	rdsAdmin	localhost	(Null)	Sleep	18	(Empty String)	(Null)
6	3805046	rdsAdmin	localhost	(Null)	Sleep	18	(Empty String)	(Null)

  - **Id**: Thread ID. You can use **kill** *id* to terminate a thread.
  - **User**: User used for connecting to the instance.
  - **Host**: IP address and port of the host that connects to the instance.
  - **db**: Database name.
  - **Command**: Connection status, which is usually **Sleep**, **Query**, or **Connect**.
  - **Time**: Connection duration, in seconds.
  - **State**: Status of the SQL statement being executed.
  - **Info**: SQL statement that is being executed.
  - CPU_time: Amount of time for which the current connection has been established.

- On the **Instances** page, locate the instance and click **View Metrics** in the **Operation** column.

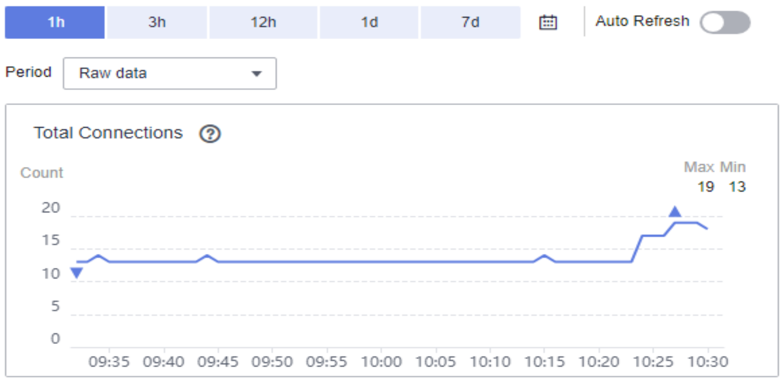
  Figure 3-19 Viewing metrics

	Name/ID	Descrip...	DB Instanc...	DB Engi...	Status	Billing ...	Private IP A...	Private IP A...	Proxy Addr...	Private Do...	Enterprise ...	Created	Databas	Operation
	gauss-b3e7 8ee097903e3143e0b58d3...	--	Primary/Sta... 2 vCPUs | 8...	GaussDB(fo...	Available	Pay-per-use	192.168...	192.168...	--	--	--	Jul 17, 2024...		Log In View Metric More

View **Total Connections**. Generally, a primary/standby instance occupies two connections. If there are more than two connections, the instance is being connected and used by other users.

Figure 3-20 Viewing total connections



3.10 How Do I Enable Availability Detection for a Connection Pool in the Service Code?

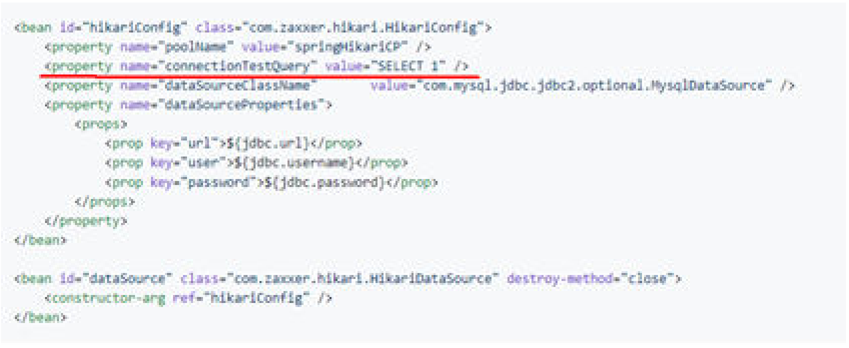
To ensure that your application obtains an available connection from a connection pool, you need to configure how the connection pool will check connection availability.

- For a JDBC or Druid connection pool:

  Set **testOnBorrow** to **true**.

- For a HikariCP connection pool:

  Set **connectionTestQuery** to **"SELECT 1"**.



```
<bean id="hikariConfig" class="com.zaxxer.hikari.HikariConfig">
    <property name="poolName" value="springHikariCP" />
    <property name="connectionTestQuery" value="SELECT 1" />
    <property name="dataSourceClassName"       value="com.mysql.jdbc.jdbc2.optional.MysqlDataSource" />
    <property name="dataSourceProperties">
        <props>
            <prop key="url">${jdbc.url}</prop>
            <prop key="user">${jdbc.username}</prop>
            <prop key="password">${jdbc.password}</prop>
        </props>
    </property>
</bean>

<bean id="dataSource" class="com.zaxxer.hikari.HikariDataSource" destroy-method="close">
    <constructor-arg ref="hikariConfig" />
</bean>
```

4 Client Installation

4.1 How Can I Install the MySQL Client?

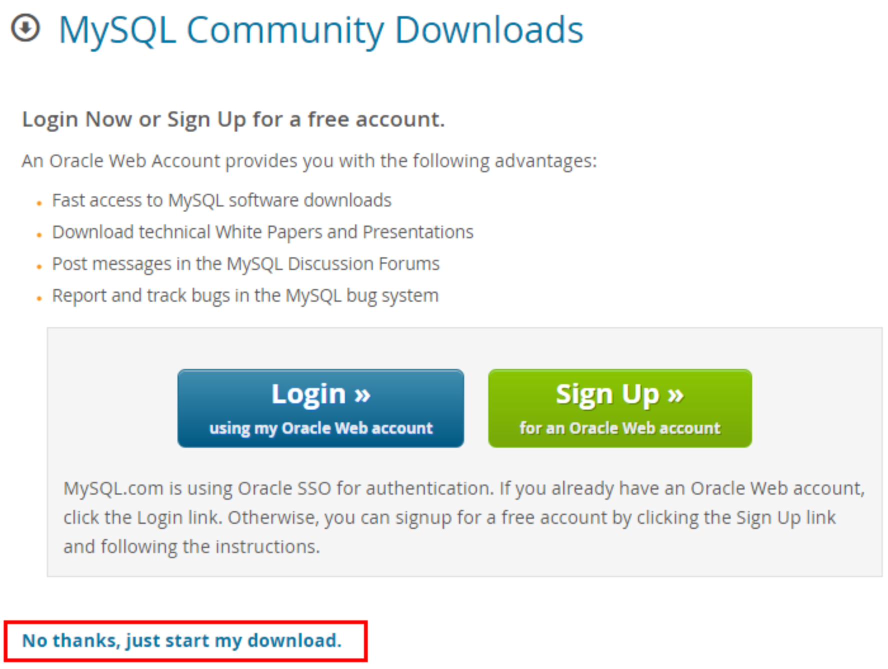
MySQL provides client installation packages for different OSs on its official website. Download the **MySQL 8.0 client installation package** or **packages of other versions**. The following uses Red Hat Linux as an example to show how to obtain the required installation package and install it.

Procedure

Step 1 Obtain the installation package.

Find the **link** to the required version on the download page. The mysql-community-client-8.0.21-1.el6.x86_64 is used as an example.

Figure 4-1 Download



MySQL Community Downloads

Login Now or Sign Up for a free account.

An Oracle Web Account provides you with the following advantages:

- Fast access to MySQL software downloads
- Download technical White Papers and Presentations
- Post messages in the MySQL Discussion Forums
- Report and track bugs in the MySQL bug system

MySQL.com is using Oracle SSO for authentication. If you already have an Oracle Web account, click the Login link. Otherwise, you can signup for a free account by clicking the Sign Up link and following the instructions.

No thanks, just start my download.

NOTE

Click **No thanks, just start my download.** to download the installation package.

Step 2 Upload the installation package to the ECS.

NOTE

When you create an ECS, select an OS, such as Red Hat 6.6, and bind an EIP to it. Then, upload the installation package to the ECS using a remote connection tool, and use PuTTY to connect to the ECS.

Step 3 Run the following command to install the MySQL client:

```
sudo rpm -ivh mysql-community-client-8.0.21-1.el6.x86_64.rpm
```

NOTE

- If any conflicts occur during the installation, add the **replacefiles** parameter to the command and try to install the client again. Example:
  ```
  rpm -ivh --replacefiles mysql-community-client-8.0.21-1.el6.x86_64.rpm
  ```
- If a message is displayed prompting you to install a dependency package, you can add the **nodeps** parameter to the command and install the client again. Example:
  ```
  rpm -ivh --nodeps mysql-community-client-8.0.21-1.el6.x86_64.rpm
  ```

Step 4 Use the MySQL client to connect to the database and check whether the client can run properly.

mysql -h <*hostIP*> **-P** <*port*> **-u** <*userName*> **-p --ssl-ca=**<*cafile*>

Table 4-1 Parameter description

Parameter	Description
<*hostIP*>	Private IP address. To obtain this parameter, go to the **Basic Information** page of the instance and view the private IP address in the **Network Information** area.
<*port*>	Database port. By default, the value is **3306**. To obtain this parameter, go to the **Basic Information** page of the instance and view the database port in the **Network Information** area.
<*userName*>	Username, that is, the GaussDB(for MySQL) database administrator account. The default value is **root**.
<*cafile*>	SSL certificate file, which should be stored in the same directory where the command is executed.

Example:

To connect to a DB instance through an SSL connection as user **root**, run the following command:

mysql -h 172.xx.xx.xx -P 3306 -u root -p --ssl-ca=ca.pem

Enter the password of the database account as prompted.

```
Enter password:
```

NOTE

If error information similar to "mysql: error while loading shared libraries: lib*xxxx*: cannot open shared object file: No such file or directory" is displayed, perform the following steps:

For example, if the error "mysql: error while loading shared libraries: libtinfo.so.5: cannot open shared object file: No such file or directory" is displayed:

1. Query the current version file of the dynamic library that reports the error on the local host.

   find / -name libtinfo.so*

   Assume that the query result is as follows:

   /usr/lib64/libtinfo.so.6.2

   /usr/lib64/libtinfo.so.6
2. Set up the soft link of the required version.

   ln -s /usr/lib64/libtinfo.so.6 /usr/lib64/libtinfo.so.5
3. Connect to the database again.

   mysql -h <*hostIP*> **-P** <*port*> **-u** <*userName*> **-p --ssl-ca=**<*cafile*>

----End

5 Database Migration

5.1 What Types of DB Engines Does GaussDB(for MySQL) Support for Importing Data?

- Exporting or importing data between DB engines of the same type is called homogeneous database export or import.
- Exporting or importing data between DB engines of different types is called heterogeneous database export or import. For example, import data from Oracle to the DB engines supported by GaussDB(for MySQL).

  Generally, data cannot be exported or imported between heterogeneous databases due to the different data formats involved. However, if the data formats are compatible, table data can, in theory, be migrated between them.

  Third-party software is usually required for data replication to export and import between heterogeneous databases. For example, you can use a third-party tool to export table records from Oracle in .txt format. Then, you can use Load statements to import the exported table records to the DB engines supported by GaussDB(for MySQL).

6 Database Permissions

6.1 Does GaussDB(for MySQL) Provide the Root Account or Super Permissions?

GaussDB(for MySQL) provides the administrator user **root** which has the permissions except super, file, shutdown, and create tablespace.

Most cloud database service platforms do not provide super permissions for the **root** user. That's because super permissions allow you to execute management commands, such as **reset master**, **set global**, **kill**, and **reset slave**. These operations may cause unpredictable errors in GaussDB(for MySQL).

If you need to perform operations that require super permissions, GaussDB(for MySQL) provides alternative methods.

- You can modify parameter values only on the GaussDB(for MySQL) console. You cannot run the following command on a database to modify parameter values.

  set global *parameter_name*=*parameter_value*;

  If the script contains the **set global** command, delete the **set global** command and modify parameter values on the GaussDB(for MySQL) console.
- An error is reported after you run the following command because the **root** user does not have super permissions. You can delete **definer='root'** from the command.

  create definer='root'@'%' trigger(procedure)...

  You can import and export data using mysqldump. For details, see **Migrating Data to GaussDB(for MySQL) Using mysqldump**.

7 Database Performance

7.1 What Should I Do If the CPU Usage of My GaussDB(for MySQL) Instance Is High?

If the CPU usage is high or close to 100% when you use GaussDB(for MySQL), data read/write processing slows down, connections cannot be established, and errors are reported, interrupting services.

Solution

1. Check slow SQL logs for slow queries and examine their performance characteristics (if any) to locate the cause.

   For details on viewing MySQL logs, see **Viewing Slow Query Logs**.
2. View the CPU usage of your GaussDB(for MySQL) instance to facilitate problem locating.

   For details, see **Introducing GaussDB(for MySQL) Metrics**.
3. Create read replicas to offload read pressure from the primary node.
4. Add indexes for associated fields in multi-table association queries.
5. Do not use the SELECT statement to scan all tables. You can specify fields or add the WHERE condition.

7.2 How Do I Handle Slow SQL Statements Caused by Inappropriate Composite Index Settings?

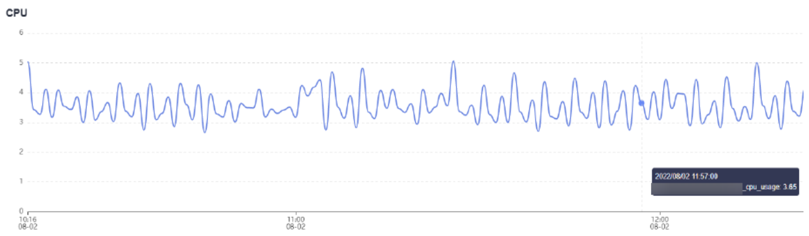
Scenario

On your instance, an SQL query that ran at 11:00 and was expected to take 8 seconds took more than 30 seconds.

Possible Causes

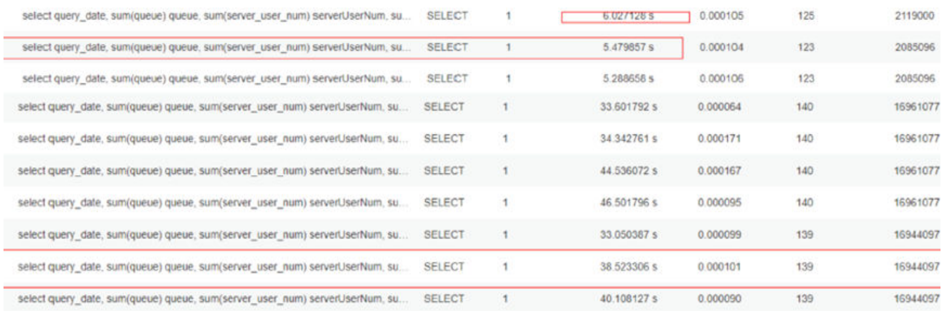
1. Check the CPU usage. In this example, during that time period, the CPU usage of the instance did not increase sharply and remained low, so we know that the slow query was not caused by high CPU usage.

   Figure 7-1 CPU usage

2. Analyze slow query logs generated during that period. In this example, shown below, there were several SQL statements that involved millions of rows being scanned. These were the slow statements. But no large amount of data was inserted into the table during that time, so we know that the slow execution was caused by missing or incorrect index settings. By running **EXPLAIN**, you can find that the execution plan of the SQL statement was full table scanning.

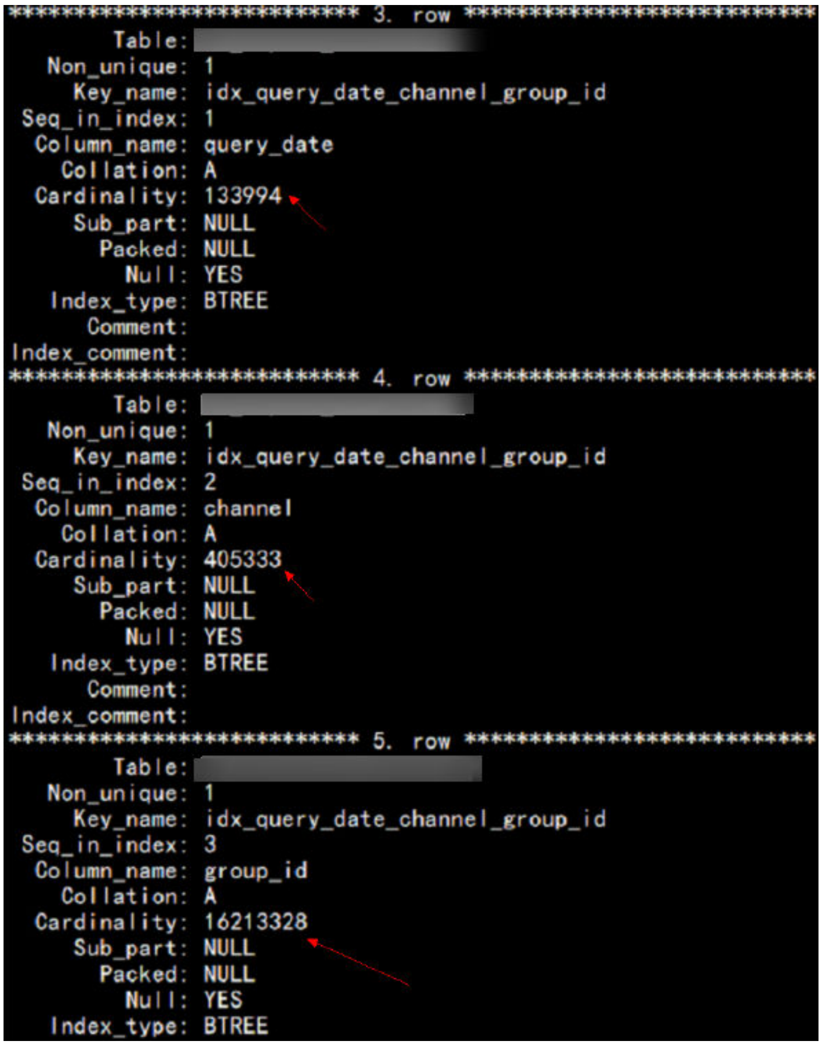
   Figure 7-2 Slow query logs

select query_date, sum(queue) queue, sum(server_user_num) serverUserNum, su...	SELECT	1	6.027128 s	0.000105	125	2119000
select query_date, sum(queue) queue, sum(server_user_num) serverUserNum, su...	SELECT	1	5.479857 s	0.000104	123	2085096
select query_date, sum(queue) queue, sum(server_user_num) serverUserNum, su...	SELECT	1	5.288658 s	0.000106	123	2085096
select query_date, sum(queue) queue, sum(server_user_num) serverUserNum, su...	SELECT	1	33.601792 s	0.000064	140	16961077
select query_date, sum(queue) queue, sum(server_user_num) serverUserNum, su...	SELECT	1	34.342761 s	0.000171	140	16961077
select query_date, sum(queue) queue, sum(server_user_num) serverUserNum, su...	SELECT	1	44.536072 s	0.000167	140	16961077
select query_date, sum(queue) queue, sum(server_user_num) serverUserNum, su...	SELECT	1	46.501796 s	0.000095	140	16961077
select query_date, sum(queue) queue, sum(server_user_num) serverUserNum, su...	SELECT	1	33.050387 s	0.000099	139	16944097
select query_date, sum(queue) queue, sum(server_user_num) serverUserNum, su...	SELECT	1	38.523306 s	0.000101	139	16944097
select query_date, sum(queue) queue, sum(server_user_num) serverUserNum, su...	SELECT	1	40.108127 s	0.000090	139	16944097

3. Perform **SHOW INDEX FROM** on the table on the instance to check the cardinality of the three columns.

Figure 7-3 Index cardinality



```
*************************** 3. row ***************************
        Table: [redacted]
   Non_unique: 1
     Key_name: idx_query_date_channel_group_id
 Seq_in_index: 1
  Column_name: query_date
    Collation: A
  Cardinality: 133994
     Sub_part: NULL
       Packed: NULL
         Null: YES
   Index_type: BTREE
      Comment:
Index_comment:
*************************** 4. row ***************************
        Table: [redacted]
   Non_unique: 1
     Key_name: idx_query_date_channel_group_id
 Seq_in_index: 2
  Column_name: channel
    Collation: A
  Cardinality: 405333
     Sub_part: NULL
       Packed: NULL
         Null: YES
   Index_type: BTREE
      Comment:
Index_comment:
*************************** 5. row ***************************
        Table: [redacted]
   Non_unique: 1
     Key_name: idx_query_date_channel_group_id
 Seq_in_index: 3
  Column_name: group_id
    Collation: A
  Cardinality: 16213328
     Sub_part: NULL
       Packed: NULL
         Null: YES
   Index_type: BTREE
```

The **query_date** field with the smallest cardinality was in the first place of the composite index, and the **group_id** field with the largest cardinality was in the last place of the composite index. In addition, the SQL statement contained the range query of the **query_date** field. As a result, only the **query_date** field was indexed.

The SQL statement could only use the index of the **query_date** column. Additionally, the optimizer may have selected full table scanning during cost estimation because the cardinality was too small.

A new composite index was created with the **group_id** field in the first place and the **query_date** field in the last place. The query time met the expectation.

Solution

1. Check whether the slow query was caused by insufficient CPU resources.
2. Check whether the table structure is properly designed and whether index settings are correct.

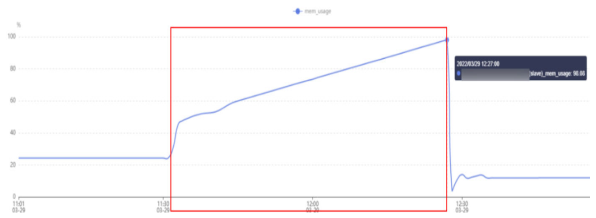
3. Execute the **ANALYZE TABLE** statement periodically to prevent incorrect execution plans because performing a large number of INSERT or DELETE operations for table data may result in outdated statistics.

7.3 How Do I Handle a Large Number of Temporary Tables Being Generated for Long Transactions and High Memory Usage?

Scenario

The memory usage of a GaussDB(for MySQL) instance kept increasing from 11:30 to 12:27 and reached the memory limit.

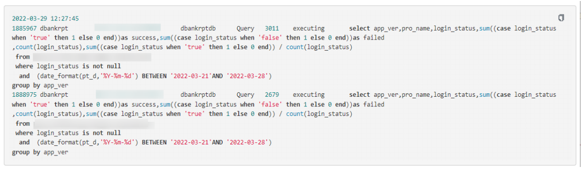
Figure 7-4 Memory usage



Possible Causes

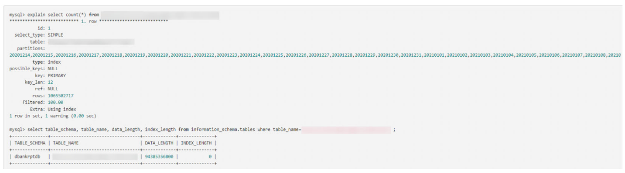
1. Check the **processlist.log** file. In this example, shown below, there were two slow SQL statements in that time period.

   Figure 7-5 Slow SQL statements

```
2022-03-29 12:27:45
1885967 dbankrpt        [redacted]        dbankrptdb      Query   3011    executing       select app_ver,pro_name,login_status,sum((case login_status
when 'true' then 1 else 0 end))as success,sum((case login_status when 'false' then 1 else 0 end))as failed
,count(login_status),sum((case login_status when 'true' then 1 else 0 end)) / count(login_status)
 from [redacted]
 where login_status is not null
  and  (date_format(pt_d,'%Y-%m-%d') BETWEEN '2022-03-21'AND '2022-03-28')
group by app_ver
1888975 dbankrpt        [redacted]        dbankrptdb      Query   2679    executing       select app_ver,pro_name,login_status,sum((case login_status
when 'true' then 1 else 0 end))as success,sum((case login_status when 'false' then 1 else 0 end))as failed
,count(login_status),sum((case login_status when 'true' then 1 else 0 end)) / count(login_status)
 from [redacted]
 where login_status is not null
  and  (date_format(pt_d,'%Y-%m-%d') BETWEEN '2022-03-21'AND '2022-03-28')
group by app_ver
```

2. Analyze slow query logs generated in that time period. There was about 90 GB of data and about 1 billion of data rows in the logs, and there were two SQL statements that took 40 to 50 minutes to execute. The execution time basically overlapped when the memory usage went up in the monitoring results, so we know that the high memory usage was caused by temporary tables.

```
mysql> explain select count(*) from [redacted]
*************************** 1. row ***************************
           id: 1
  select_type: SIMPLE
        table: [redacted]
   partitions:
20201214,20201215,20201216,20201217,20201218,20201219,20201220,20201221,20201222,20201223,20201224,20201225,20201226,20201227,20201228,20201229,20201230,20201231,20210101,20210102,20210103,20210104,20210105,20210106,20210107,20210108,20210
         type: index
possible_keys: NULL
          key: PRIMARY
      key_len: 12
          ref: NULL
         rows: 1065502717
     filtered: 100.00
        Extra: Using index
1 row in set, 1 warning (0.00 sec)

mysql> select table_schema, table_name, data_length, index_length from information_schema.tables where table_name=[redacted] ;
+--------------+-----------------------------------+-------------+--------------+
| TABLE_SCHEMA | TABLE_NAME                        | DATA_LENGTH | INDEX_LENGTH |
+--------------+-----------------------------------+-------------+--------------+
| dbankrptdb   | [redacted]                        | 94385356800 |            0 |
+--------------+-----------------------------------+-------------+--------------+
```

Solution

1. Upgrade the instance specifications to maintain the memory usage within a proper range, preventing a sudden increase in traffic from causing an OOM crash. For details, see **Changing vCPUs and Memory of a DB Instance**.

2. Optimize slow SQL statements as needed.

7.4 What Should I Do If Locks on Long Transactions Block the Execution of Subsequent Transactions?

Scenario

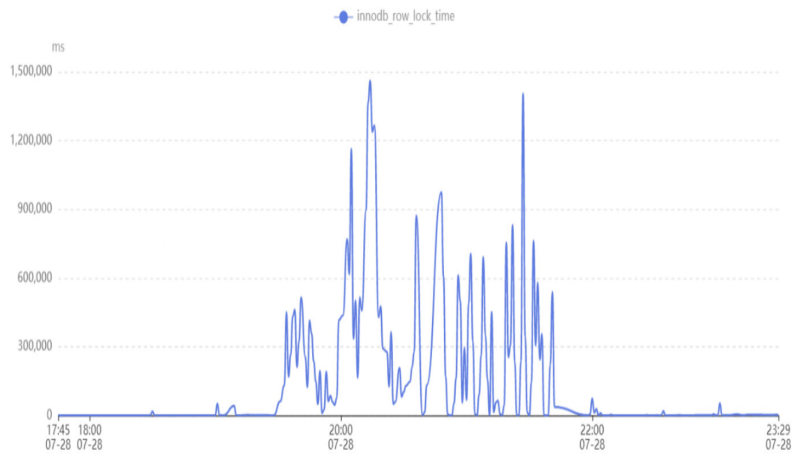
Error code 1205 was reported:

"MySQL error code MY-001205 (ER_LOCK_WAIT_TIMEOUT): **Lock wait timeout exceeded**; try restarting transaction"

Possible Causes

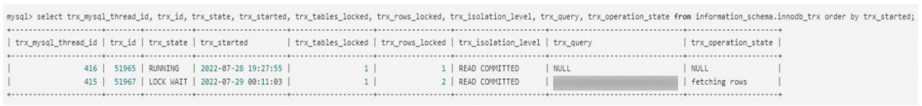
1. Check the value of the monitoring metric **Row Lock Time**. In this example, the value of this metric was high, so we know there were lock conflicts in the system.

   For details about monitoring metrics, see **Viewing Instance Monitoring Metrics**.



2. Log in to the DB instance and run the following SQL statement to check the long transactions in the system and the row locks held by the transactions:

```
select trx_mysql_thread_id, trx_id, trx_state, trx_started, trx_tables_locked, trx_rows_locked,
trx_isolation_level, trx_query, trx_operation_state from information_schema.innodb_trx order by
trx_started;
```



```
mysql> select trx_mysql_thread_id, trx_id, trx_state, trx_started, trx_tables_locked, trx_rows_locked, trx_isolation_level, trx_query, trx_operation_state from information_schema.innodb_trx order by trx_started;
+---------------------+--------+-----------+---------------------+-------------------+-----------------+---------------------+-----------------------------+---------------------+
| trx_mysql_thread_id | trx_id | trx_state | trx_started         | trx_tables_locked | trx_rows_locked | trx_isolation_level | trx_query                   | trx_operation_state |
+---------------------+--------+-----------+---------------------+-------------------+-----------------+---------------------+-----------------------------+---------------------+
|                 416 |  51965 | RUNNING   | 2022-07-28 19:27:55 |                 1 |               1 | READ COMMITTED      | NULL                        | NULL                |
|                 415 |  51967 | LOCK WAIT | 2022-07-29 00:11:03 |                 1 |               2 | READ COMMITTED      | [redacted]                  | fetching rows       |
+---------------------+--------+-----------+---------------------+-------------------+-----------------+---------------------+-----------------------------+---------------------+
```

   – **information_schema.innodb_trx**: information about transactions that are being executed in the InnoDB.

   – **trx_started**: start time of a transaction, which is used to determine whether the current transaction is a long transaction. The execution time of a transaction is the current time minus the start time.

   – **trx_state**: Status of the current transaction. The values are as follows:

      ■ **RUNNING**

- **LOCK WAIT**

  NOTE

  If the status of a transaction is **LOCK WAIT**, the transaction holds a row lock.

- **ROLLING BACK**
- **COMMITTING**

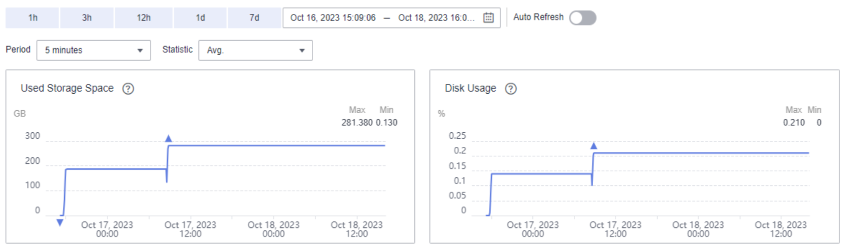
Solution

Kill the long transactions.

7.5 How Can I Use Temporary Disk of GaussDB(for MySQL)?

Temporary disks of GaussDB(for MySQL) instances are used to temporarily store temporary tables, temporary files, and binlog caches generated during database operation. On the management console, you can monitor used temporary disk space and temporary disk usage of your instance in different time periods and granularities in real time, as shown in the following figure.

Figure 7-6 Temporary disk usage



As services fluctuate, you may find that the usage of temporary disks suddenly or continuously increases. To improve database availability and stability, GaussDB(for MySQL) provides up to 500 GB of temporary disk space for a DB instance for free.

To prevent the temporary disk usage from continuously increasing and ultimately getting filled up, you are advised to check services as soon as possible based on the queried disk usage. This section describes the risks, scenarios, and troubleshooting you can perform when temporary disks are full.

Risks

- SQL statements fail to be executed and no results are returned.
- SQL statements occupy lock resources for a long time and block other SQL statements. As a result, the number of connections increases or even reaches the upper limit, affecting other services.

- There are too many temporary files in the binlog cache, which can cause the database to break down. This takes a long time to restore, so services are interrupted for a long time.

Scenarios and Troubleshooting

1. Explicitly creating temporary disk tables
   - Scenario

     You can run the **create temporary table** statement to explicitly create temporary disk tables. The temporary tables whose storage engine is InnoDB are cached in the buffer pool and flushed to disks by dirty threads.

     In GaussDB(for MySQL), data in disk temporary tables is stored in session temporary tablespace (the path is specified by the **innodb_temp_tablespaces_dir** parameter), and undo logs are stored in the global temporary tablespace (the path is specified by the **innodb_temp_data_file_path** parameter).

     To prevent temporary disk tables from occupying too much disk space, you are advised to delete unnecessary temporary disk tables or disconnect unnecessary database connections.

     NOTICE

     - Session temporary tablespace: It is reclaimed when the current database connection is released.
     - Global temporary tablespace: It is reclaimed only after the database is restarted.

   - Troubleshooting
     i. View information about the temporary tables you created in InnoDB.

```
mysql> select * from information_schema.innodb_temp_table_info;
+----------------------+---------------+--------+------------+
| TABLE_ID             | NAME          | N_COLS | SPACE      |
+----------------------+---------------+--------+------------+
| 18446744069414584311 | #sqle055_24_0 |      5 | 4294502266 |
+----------------------+---------------+--------+------------+
```

     ii. Check the usage of InnoDB temporary table files.

     In a table, the **ID** column indicates the ID of the session that is using the temporary table file. If the value is **0**, the ibt file is not used. The **SIZE** column indicates the size of the ibt file, which automatically increases based on the usage and is reclaimed when the session ends. If the value of the **PURPOSE** column is **INTRINSIC**, the table is an implicit temporary table. If the value of the **PURPOSE** column is **USER**, the table is an explicit temporary table.

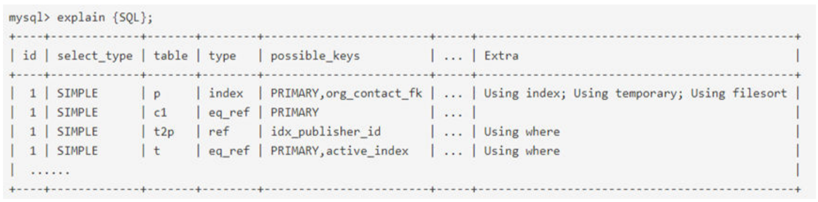
```
mysql> select * from information_schema.innodb_session_temp_tablespaces;
+----+------------+----------------------------+-------+----------+-----------+
| ID | SPACE      | PATH                       | SIZE  | STATE    | PURPOSE   |
+----+------------+----------------------------+-------+----------+-----------+
| 31 | 4294502265 | ./#innodb_temp/temp_9.ibt  | 81920 | ACTIVE   | INTRINSIC |
| 36 | 4294502266 | ./#innodb_temp/temp_10.ibt | 98304 | ACTIVE   | USER      |
| 34 | 4294502264 | ./#innodb_temp/temp_8.ibt  | 81920 | ACTIVE   | INTRINSIC |
|  0 | 4294502257 | ./#innodb_temp/temp_1.ibt  | 81920 | INACTIVE | NONE      |
|  0 | 4294502258 | ./#innodb_temp/temp_2.ibt  | 81920 | INACTIVE | NONE      |
|  0 | 4294502259 | ./#innodb_temp/temp_3.ibt  | 81920 | INACTIVE | NONE      |
|  0 | 4294502260 | ./#innodb_temp/temp_4.ibt  | 81920 | INACTIVE | NONE      |
|  0 | 4294502261 | ./#innodb_temp/temp_5.ibt  | 81920 | INACTIVE | NONE      |
|  0 | 4294502262 | ./#innodb_temp/temp_6.ibt  | 81920 | INACTIVE | NONE      |
|  0 | 4294502263 | ./#innodb_temp/temp_7.ibt  | 81920 | INACTIVE | NONE      |
+----+------------+---------------------------+-------+----------+-----------+
```

2. Querying disk temporary tables or temporary files implicitly created
   - Scenario

     When selecting an execution plan for a query, the query optimizer may use temporary tables. Temporary memory tables are preferentially used. When the size of temporary memory tables exceeds a certain threshold (either **tmp_table_size** or **max_heap_table_size**, whichever is smaller), temporary disk tables are used.

     Disk temporary tables are implicitly created by queries. The data between tables that are implicitly and explicitly created are the same and stored in the session temporary tablespace. If there are complex queries, including but not limited to keywords such as UNION, GROUP BY, and ORDER BY, in larger tables, temporary disk tables may be generated. In addition, when queries involve sorting operations, if the sort buffer cannot store all of the data (the buffer size is specified by **sort_buffer_size**), temporary disk files can be used for auxiliary sorting. In most scenarios, temporary disk tables implicitly created are the main reason disks fill up. If your disk is filling up, you can locate complex queries or long transactions, optimize query statements, add proper indexes, and split long transactions to address the issue.
   - Troubleshooting
     1. Check whether there are SQL statements using temporary tables or file sorting.

        If **Using temporary** is displayed in the **Extra** column, temporary tables are used. If **Using filesort** is displayed, file sorting is used.

        ```
        mysql> explain {SQL};
        +----+-------------+-------+--------+----------------------+-----+---------------------------------------------+
        | id | select_type | table | type   | possible_keys        | ... | Extra                                       |
        +----+-------------+-------+--------+----------------------+-----+---------------------------------------------+
        |  1 | SIMPLE      | p     | index  | PRIMARY,org_contact_fk | ... | Using index; Using temporary; Using filesort |
        |  1 | SIMPLE      | c1    | eq_ref | PRIMARY              | ... |                                             |
        |  1 | SIMPLE      | t2p   | ref    | idx_publisher_id     | ... | Using where                                 |
        |  1 | SIMPLE      | t     | eq_ref | PRIMARY,active_index | ... | Using where                                 |
        |  ......                                                                                                      |
        +----+-------------+-------+--------+----------------------+-----+---------------------------------------------+
        ```

     2. Query the usage of implicit temporary tables. The method is the same as that the explicit disk temporary tables.
3. Querying binlogs generated for long transactions
   - Scenario

     A binlog is a binary that records database changes, such as DDL, DCL, and DML (excluding SELECT). InnoDB caches binlogs in the memory before transactions are committed and writes binlogs to disks only after the transactions are committed. The size of the binlog file for each connection in the memory is specified by the **binlog_cache_size** parameter. When the size of the binlog file recorded by a transaction exceeds the value of this parameter, the binlog file is written to a temporary disk file. Long transactions may cause large binlogs. As a result, the size of temporary binlogs on the disk is large and the disk may be full. You are advised to control the transaction size, split long transactions, or change **binlog_cache_size** to a more appropriate value.

– Troubleshooting

i. Check whether binlog is enabled.

```
mysql> show variables like 'log_bin';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| log_bin       | ON    |
+---------------+-------+
```

ii. View the binlog cache usage.

Binlog_cache_disk_use indicates the number of times that temporary disk files are used for caching binlogs due to insufficient memory (specified by **binlog_cache_size**). If the value of **binlog_cache_size** is large, temporary disk files are invoked for caching binlogs multiple times.

```
mysql> show global status like '%binlog_cache%';
+-----------------------+-----------+
| Variable_name         | Value     |
+-----------------------+-----------+
| Binlog_cache_disk_use | 1335006   |
| Binlog_cache_use      | 264240359 |
+-----------------------+-----------+
```

4. Checking temporary files generated by DDLs

– Scenario

During DDL operations on tables, temporary disk files are generated in some phases.

▪ Sometimes, you need to re-create the tablespace of the original table, which involves the re-creation of the B+ tree index on the table. If a table contains a large amount of data, the sort buffer cannot store all of the data. You need to create a temporary file to assist with the sorting.

▪ Although some online DDL statements support DML operations on the original table, the original table cannot be directly modified. The modification must be recorded in online logs and applied to the new table after the DDL operations are complete. Online logs are preferentially stored in the memory. The size of online logs is specified by the **innodb_sort_buffer_size** parameter. If the size of online logs exceeds the parameter value, online logs are temporarily stored in a temporary file.

▪ When the **OPTIMIZE TABLE** statement is executed on a table, the data stored in the clustered index needs to be reorganized, which may generate temporary files.

– Troubleshooting

▪ Run the **SHOW PROCESSLIST** command to check whether there are DDL statements that have been taking a long time to execute.

▪ Ensure that there is enough space before performing DDLs for large tables.

7.6 What Is the CPU Usage of a GaussDB(for MySQL) Instance with Empty Load?

A GaussDB(for MySQL) instance has the operating system process, mysqld process, monitoring process, and incremental backup process. The mysqld process contains multiple threads, such as the primary/standby communications thread, connection thread, and refresh thread. The monitoring process monitors the instance status in real time. The incremental backup process backs up incremental data. Even when an instance is unloaded, there are still multiple processes and threads running in the background, resulting in non-zero CPU usage. Typically, the CPU usage ranges from 10% to 15% in such cases.

8 Database Usage

8.1 Why Are the Results Inconsistent After the MATCH AGAINST Statement Is Executed, Respectively, on Primary Nods and Read Replicas?

MATCH GAINST is used to search for MySQL full-text indexes. For rows in the table, MATCH returns relevance values, that is, a similarity measure between the search string (given as the argument to AGAINST() function and the text in that row in the columns named in the MATCH() list). This statement uses the **stat_n_rows** value to calculate the relevance value. Primary nodes and read replicas use different methods to obtain the **stat_n_rows** value. The primary nodes use the persistent method and the read replicas use the transient method. Therefore, the obtained values are slightly different from each other. The execution result of MATCH AGAINST on primary nodes and read replicas are different.

8.2 How Do I Add Columns Using INSTANT?

GaussDB(for MySQL) is compatible with open-source MySQL 8.0.22, so you can use **ALGORITHM=INSTANT** to quickly add columns, preventing lock waiting from affecting services or SQL statement execution timeout.

Constraints

- Columns can be added only in one statement. If there are other non-INSTANT operations in the same statement, columns cannot be added immediately.
- Columns can be added only at the end of existing columns.
- COMPRESSED row format is not supported.
- Tables that already have full-text indexes are not supported.

NOTE

If a table has a full-text index, you must run the OPTIMIZE TABLE statement on the table after deleting the full-text index.

- Temporary tables are not supported.
- A new field cannot have a default value.

Procedure

Step 1 **Log in to the management console.**

Step 2 Click in the upper left corner and select a region and project.

Step 3 Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.

Step 4 On the **Instances** page, locate the instance and click **Log In** in the **Operation** column.

Alternatively, on the **Instances** page, click the instance name to go to the **Basic Information** page. Click **Log In** in the upper right corner of the page.

Step 5 On the displayed login page, enter the correct username and password and click **Log In**.

Step 6 On the top menu bar, choose **SQL Operations** > **SQL Query**.

Step 7 Run the following SQL statement to quickly add a column:

ALTER TABLE *table_name* ADD COLUMN *column_name column_definition*, ALGORITHM=INSTANT;

- *table_name*: table name
- *column_name*: column name
- *column_definition*: column remarks

----End

8.3 How Do I Use LOAD DATA to Import Local Data?

You can use LOAD DATA to import local data to GaussDB(for MySQL).

Syntax

```
LOAD DATA LOCAL
    INFILE 'file_name'
    [REPLACE | IGNORE]
    INTO TABLE tbl_name
    [CHARACTER SET charset_name]
    [{FIELDS | COLUMNS}
        [TERMINATED BY 'string']
        [[OPTIONALLY] ENCLOSED BY 'char']
    ]
    [LINES
        [TERMINATED BY 'string']
    ]
    [IGNORE number {LINES | ROWS}]
    [(col_name_or_user_var
        [, col_name_or_user_var] ...)]
```

Parameters

- **file_name**: path of the local file to be imported.
- **REPLACE | IGNORE**: whether to replace or ignore duplicate records.
- **tbl_name**: name of the table to be imported.
- **CHARACTER SET charset_name**: file encoding format. You are advised to use the encoding format of the GaussDB(for MySQL) instances to avoid garbled characters.
- **FIELDS TERMINATED BY 'string'**: separator between columns. The default value is **\t**.
- **[OPTIONALLY] ENCLOSED BY 'char'**: used to ignore symbols in data source fields.
- **LINES TERMINATED BY'string??**: newline character between lines. The default value is **\n**.

  NOTE

  On some hosts running the Windows servers, the newline characters of text files may be **\r\n**, which is invisible.

- **IGNORE number LINES**: used to ignore lines at the start of the file.
- **(column_name_or_user_var, ...)**: columns to be imported. If this parameter is not configured, data is imported based on the column sequence by default.
- For other parameters, see the **load data infile** on the MySQL official website. The sequence of other parameters must be correct. For sequence details, visit **the MySQL official website**.

Standard Example

Prerequisites

- The **local_infile** parameter must be enabled on the server. Click the instance name to go to the **Basic Information** page. On the **Parameters** page, change the value of this parameter to **ON**.
- The **local-infile** parameter must be enabled on the client. Configure **local-infile** in the **my.cnf** file or use the **--local-infile=1** option to connect to the database.

```
[mysql]
local-infile
```

1. Import the data in the local file **qq.txt** to the **test** table. The **qq.txt** file contains five rows of data. The column separator is **','** and the row separator is **'\n'**.

```
1,a
2,b
3,c
4,d
5,"e"
```

2. Create the **test** table.

```
CREATE TABLE test (
`id` int NOT NULL,
`a` varchar(4) NOT NULL,
PRIMARY KEY (`id`)
);
```

3. On the client, run the LOAD DATA statement to import data in the **qq.txt** file to the **test** table, set the character set to **utf8**, and ignore the double quotation marks in the data source field.

```
mysql> LOAD DATA LOCAL INFILE '/data/qq.txt' IGNORE INTO TABLE test CHARACTER SET 'utf8'
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"' LINES TERMINATED BY '\n';
Query OK, 5 rows affected, 1 warning (0.00 sec)
Records: 5  Deleted: 0  Skipped: 0  Warnings: 1

mysql> select * from test;
+----+---+
| id | a |
+----+---+
|  1 | a |
|  2 | b |
|  3 | c |
|  4 | d |
|  5 | e |
+----+---+
5 rows in set (0.00 sec)
```

NOTICE

1. Importing data affects performance of GaussDB(for MySQL) instances. Import data during off-peak hours.
2. Do not to initiate multiple LOAD DATA requests at the same time. When multiple LOAD DATA requests are initiated, SQL transactions may time out due to highly concurrent data write operations, table locking, and system I/O occupation, resulting in failure of all LOAD DATA requests.

8.4 How Do I Write Data to or Create Indexes for an Ultra-large Table?

Writing Data to an Ultra-large Table

For a table with tens of millions or hundreds of millions of data records, you are advised to use the following methods to improve data write efficiency:

- Delete unnecessary indexes.

  When data is updated, the index data is also updated. For a table with large amounts of data, avoid creating too many indexes as this can slow down the update process. Delete unnecessary indexes based on service evaluation.
- Use batch insertion to insert multiple data records.

  This is because batch insertion only requires a single remote request to the database.

  Example:

  ```
  insert into tb1 values(1,'value1');
  insert into tb2 values(2,'value2');
  insert into tb3 values(3,'value3');
  ```

  After optimization:

  ```
  insert into tb values(1,'value1'),(2,'value2'),(3,'value3');
  ```
- When inserting multiple data records, manually control transactions.

By manually controlling transactions, multiple execution units can be merged into a single transaction, avoiding the overhead of multiple transactions while ensuring data integrity and consistency.

Example:

```
insert into table1 values(1,'value1'),(2,'value2'),(3,'value3');
insert into table2 values(4,'value1'),(5,'value2'),(6,'value3');
insert into table3 values(7,'value1'),(8,'value2'),(9,'value3');
```

After optimization:

```
start transaction;
insert into table1 values(1,'value1'),(2,'value2'),(3,'value3');
insert into table2 values(4,'value1'),(5,'value2'),(6,'value3');
insert into table3 values(7,'value1'),(8,'value2'),(9,'value3');
commit;
```

CAUTION

Having too many merged statements can lead to large transactions, which will lock the table for a long time. Evaluate service needs and control the number of statements in a transaction accordingly.

- When inserting data, insert primary keys in sequential order. You can use AUTO_INCREMENT.

  Inserting primary keys in a random order can cause page splitting, which can negatively impact performance.

  Example:

  Inserting primary keys in a random order: 6 2 9 7 2

  Inserting primary keys in sequential order: 1 2 4 6 8
- Avoid using UUIDs or other natural keys, such as ID card numbers, as primary keys.

  UUIDs generated each time are unordered, and inserting them as primary keys can cause page splitting, which can negatively impact performance.
- Avoid modifying primary keys during service operations.

  Modifying primary keys requires modifying the index structure, which can be costly.
- Reduce the length of primary keys as much as possible if the business permits.
- Do not use foreign keys to maintain foreign key relationships. Use programs instead.
- Separate read and write operations. Place read operations on read replicas to avoid slow insertion caused by I/Os.

Creating Indexes for an Ultra-large Table

For a table with tens of millions or hundreds of millions of data records, you are advised to use the following methods to improve index creation efficiency:

- Keep the index field as small as possible.
- Select a column with high distinction as the index column.

- If each field in the table cannot guarantee uniqueness, cannot guarantee NOT NULL, or is not suitable for indexing, create a custom ID auto-increment column as the primary key, which will automatically ensure ordered insertion.
- To create an index, insert data first and then run the **alter table add index** command.
- Use GaussDB(for MySQL) Parallel DDL to create an index. When database hardware resources are idle, you can use parallel DDL to accelerate DDL execution, preventing subsequent DML operations from being blocked and shortening the DDL operation window.

8.5 What Are the Risks of Deleting an Index from an Ultra-large Table?

Deleting an index is risky. You are advised not to delete an index unless necessary. The reasons are as follows:

- Deleting an index will deteriorate the performance of queries that use the index. Slow SQL statements consume all system resources, which affects workloads.
- When an index is deleted, the table is locked and other users cannot access the table, which affects system availability.
- When an index is deleted, index data may be lost or damaged, which affects data consistency.
- If workloads are affected after an index is deleted, the index needs to be rebuilt, which is time-consuming for ultra-large tables.

9 Backups

9.1 How Long Does GaussDB(for MySQL) Store Backup Data?

Automated backup data is kept based on the backup retention period you specified. For details, see **Configuring a Same-Region Backup Policy**.

There is no limit for the manual backup retention period. For details, see **Deleting a Manual Backup**.

The backup data is stored in OBS and does not occupy the storage space of your buy instance.

9.2 How Can I Check and Clear My GaussDB(for MySQL) Backup Space?

The GaussDB(for MySQL) backup space stores automated backups and manual backups. You can check the backup space usage in the **Storage/Backup Space** area on the GaussDB(for MySQL) console and clear the backup space as needed.

Checking the GaussDB(for MySQL) Backup Space Usage

Step 1 **Log in to the management console.**

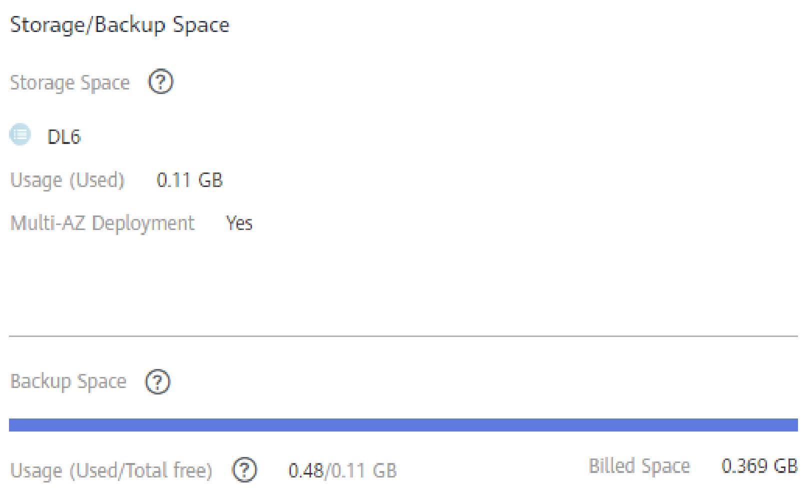
Step 2 Click in the upper left corner and select a region and project.

Step 3 Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.

Step 4 On the **Instances** page, click the instance name to go to the **Basic Information** page.

Step 5 Check the backup space usage in the **Storage/Backup Space** area.

Figure 9-1 Checking the backup space usage


Storage/Backup Space

Storage Space

DL6

Usage (Used) 0.11 GB

Multi-AZ Deployment Yes

Backup Space

Usage (Used/Total free) 0.48/0.11 GB

Billed Space 0.369 GB

NOTE

- There are full and incremental backups of a DB instance in the backup space.
- Free backup storage space of the same size as your purchased storage space are provided.

----End

Clearing the GaussDB(for MySQL) backup space

- **Automated full and incremental backups**

  Automated backups cannot be manually deleted. You need to change the backup cycle by referring to **Modifying a Same-Region Backup Policy**. The backup files that have expired will be deleted.
- **Manual full backups**

  You can manually delete manual backups. For details, see **Deleting a Manual Backup**.

9.3 How Can I Back Up a GaussDB(for MySQL) Database to an ECS?

You can back up data to an ECS in the same way you export SQL statements. The ECS service does not have restrictions on the types of data to be backed up as long as the data complies with local laws and regulations. You can store backup data on an ECS.

However, you are advised to use GaussDB(for MySQL) **automated backups** and **manual backups** to back up data to OBS for higher data reliability and service assurance.

9.4 Why Has My Automated Backup Failed?

The possible reasons for automated backup failures are as follows:

1. The network environment may be unstable due to problems such as network delay or interruptions. If GaussDB(for MySQL) detects any of these problems, it triggers another automated backup half an hour later. Alternatively, you can perform a manual backup immediately.
2. If multiple tasks are being executed simultaneously, there can be problems such as excessive task wait times or interruptions. If GaussDB(for MySQL) detects any of these problems, it triggers another automated backup half an hour later. Alternatively, you can perform a manual backup immediately.
3. The DB instance is abnormal probably because it is faulty or being modified. If GaussDB(for MySQL) detects any of these problems, it triggers an automated backup when the instance status becomes available. Alternatively, you can perform a manual backup immediately.
4. A parameter change was incorrect. If your DB instance becomes faulty after you modify parameters of a parameter template and apply the template to the instance, check whether the modified parameters are set to correct values and whether there are any associated parameters that need to be changed, or reset the parameters to their defaults and reboot the DB instance.
5. An error occurred during data import. For example, system catalog records get lost due to inappropriate data import. You can import data again by referring to Migrating Data to GaussDB(for MySQL) Using DRS. The system will automatically identify and back up the data again.
6. If the problem persists, submit a service ticket by choosing **Service Tickets > Create Service Ticket** in the upper right corner of the management console.

9.5 How Is GaussDB(for MySQL) Backup Data Billed?

All the GaussDB(for MySQL) backups are stored on OBS without occupying the storage of your DB instances. GaussDB(for MySQL) provides free backup space of the same size as your purchased storage.

For example, if you purchase a DB instance with 200 GB of storage, you can get an additional 200 GB of backup space and are only charged for backups in excess of 200 GB. The first 200 GB of backup data is free. When the 200 GB storage is used up, the backups will be billed on a pay-per-use basis. For pricing details, see **Product Pricing Details**.

NOTICE

If your storage is frozen, it is no longer charged and the free backup space is also unavailable.

If your DB instance is frozen, no free backup space is available. As a result, the original automated backups of the DB instance will be charged.

- If you unfreeze the DB instance, the free backup space will be restored.
- If you directly delete the frozen DB instance, its automated backups will also be deleted and the backup space will not be charged any longer.

10 Database Parameter Modification

10.1 How Do I Change the Time Zone of a GaussDB(for MySQL) Instance?

GaussDB(for MySQL) allows you to select a time zone when you create an instance and change the time zone after the instance is created.

Procedure

Step 1 **Log in to the management console.**

Step 2 Click in the upper left corner and select a region and project.

Step 3 Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.

Step 4 On the **Instances** page, click the instance name.

Step 5 In the navigation pane, choose **Parameters**.

Step 6 Search for a time zone parameter in the search box, for example, **time_zone**.

Step 7 Select a time zone, and click **Save**.

Step 8 In the displayed dialog box, click **Yes**.

For example, to change the time zone to UTC+08:00, select **Asia/Shanghai** from the drop-down list.

----End

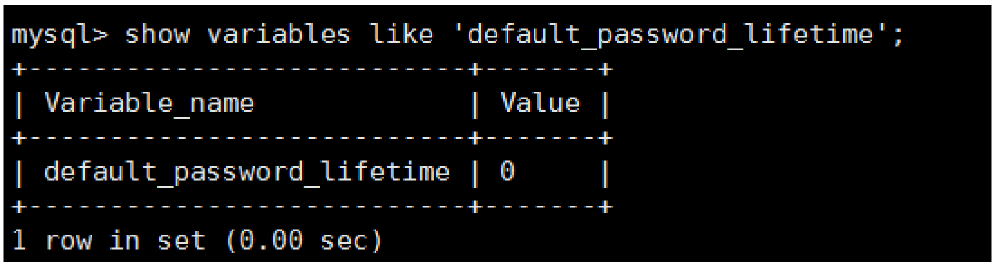
Time Zone Parameters

- **system_time_zone**: operating system (OS) time zone. Changing the value of this parameter does not affect the database time zone.
- **time_zone**: database time zone. You can modify this parameter to change the time zone for your instance.

10.2 How Do I Configure a Password Expiration Policy for GaussDB(for MySQL) Instances?

In GaussDB(for MySQL) 8.0, you can set the global variable **default_password_lifetime** to control the default validity period of a user password.

The value of **default_password_lifetime** indicates how many days until a password expires. The default value is **0**, indicating that the created user password will never expire.

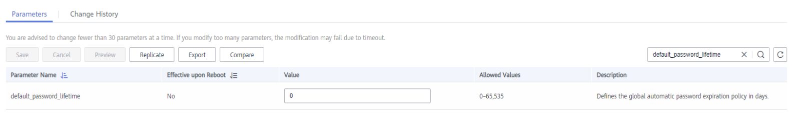


```
mysql> show variables like 'default_password_lifetime';
+---------------------------+-------+
| Variable_name             | Value |
+---------------------------+-------+
| default_password_lifetime | 0     |
+---------------------------+-------+
1 row in set (0.00 sec)
```

Changing the Global Automatic Password Expiration Policy

You can change the global automatic password expiration policy in either of the following ways:

- Change the value of **default_password_lifetime** on the GaussDB(for MySQL) console.
  a. **Log in to the management console.**
  b. Click in the upper left corner and select a region and project.
  c. Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.
  d. On the **Instances** page, click the instance name.
  e. In the navigation pane, choose **Parameters**.
  f. Search for **default_password_lifetime** in the search box.



You are advised to change fewer than 30 parameters at a time. If you modify too many parameters, the modification may fail due to timeout.

default_password_lifetime

Parameter Name	Effective upon Reboot	Value	Allowed Values	Description
default_password_lifetime	No	0	0–65,535	Defines the global automatic password expiration policy in days.

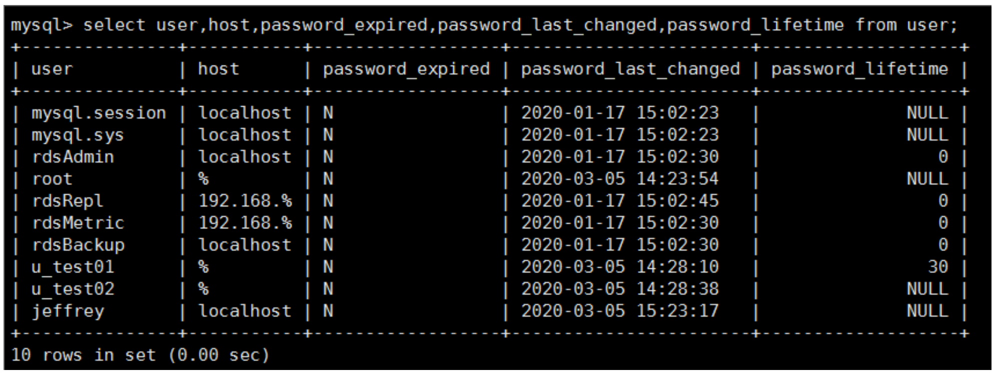
  g. Change its value and click **Save**.
  h. In the displayed dialog box, click **Yes**.
- Run the following command to change the value of **default_password_lifetime**:

  set global default_password_lifetime=0;

Checking the Password Expiration Date of All Users

Run the following command:

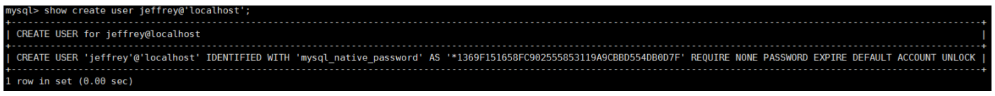
select user,host,password_expired,password_last_changed,password_lifetime from user;


```
mysql> select user,host,password_expired,password_last_changed,password_lifetime from user;
+---------------+-----------+------------------+-----------------------+-------------------+
| user          | host      | password_expired | password_last_changed | password_lifetime |
+---------------+-----------+------------------+-----------------------+-------------------+
| mysql.session | localhost | N                | 2020-01-17 15:02:23   |              NULL |
| mysql.sys     | localhost | N                | 2020-01-17 15:02:23   |              NULL |
| rdsAdmin      | localhost | N                | 2020-01-17 15:02:30   |                 0 |
| root          | %         | N                | 2020-03-05 14:23:54   |              NULL |
| rdsRepl       | 192.168.% | N                | 2020-01-17 15:02:45   |                 0 |
| rdsMetric     | 192.168.% | N                | 2020-01-17 15:02:30   |                 0 |
| rdsBackup     | localhost | N                | 2020-01-17 15:02:30   |                 0 |
| u_test01      | %         | N                | 2020-03-05 14:28:10   |                30 |
| u_test02      | %         | N                | 2020-03-05 14:28:38   |              NULL |
| jeffrey       | localhost | N                | 2020-03-05 15:23:17   |              NULL |
+---------------+-----------+------------------+-----------------------+-------------------+
10 rows in set (0.00 sec)
```

Checking the Password Expiration Policy of a Specified User

Run the following command:

show create user *jeffrey*@'localhost';


```
mysql> show create user jeffrey@'localhost';
+----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| CREATE USER for jeffrey@localhost                                                                                                                                    |
+----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| CREATE USER 'jeffrey'@'localhost' IDENTIFIED WITH 'mysql_native_password' AS '*1369F151658FC902555853119A9CBBD554DB0D7F' REQUIRE NONE PASSWORD EXPIRE DEFAULT ACCOUNT UNLOCK |
+----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

EXPIRE DEFAULT indicates that the password follows the global expiration policy.

Configuring the Password Expiration Policy for a Specified User

- Configuring the password expiration policy during user creation

  create user '*script*'@'localhost' identified by '*******' password expire interval 90 day;**

- Configuring the password expiration policy after user creation

  ALTER USER '*script*'@'localhost' PASSWORD EXPIRE INTERVAL 90 DAY;

- Setting the password to be permanently valid

  CREATE USER '*mike*'@'%' PASSWORD EXPIRE NEVER;

  ALTER USER '*mike*'@'%' PASSWORD EXPIRE NEVER;

- Setting the password to follow the global expiration policy

  CREATE USER '*mike*'@'%' PASSWORD EXPIRE DEFAULT;

  ALTER USER '*mike*'@'%' PASSWORD EXPIRE DEFAULT;

10.3 How Do I Ensure that the Database Character Set of a GaussDB(for MySQL) Instance Is Correct?

UTF-8 supports 4 byte characters, but GaussDB(for MySQL) utf8 supports only 3 byte characters. Emojis and newly added Unicode characters cannot be stored using MySQL utf8 character set. MySQL released the utf8mb4 character set in 2010 and added the utf8mb4 code after 5.5.3 to be compatible with the 4-byte unicode. You only need to change utf8 to utf8mb4. No other conversion is required.

Huawei Cloud Data Admin Service (DAS) is a professional database management tool. You can view the database and system character sets through the DAS console.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner and select a region and project.

Step 3 Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.

Step 4 On the **Instances** page, locate the instance and click **Log In** in the **Operation** column.

Alternatively, on the **Instances** page, click the instance name to go to the **Basic Information** page. Click **Log In** in the upper right corner of the page.

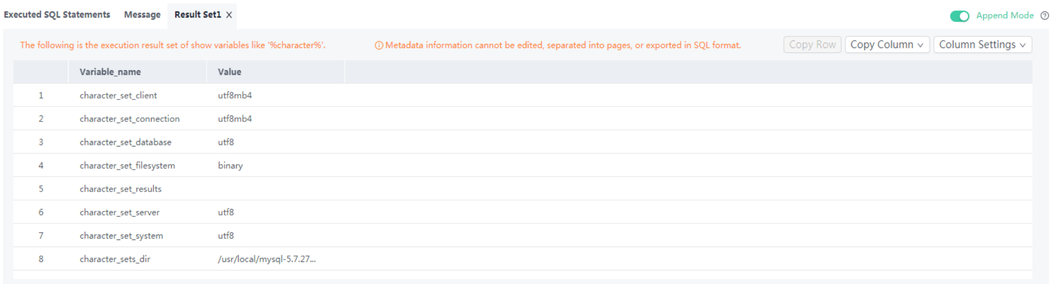
Step 5 On the displayed login page, enter the correct username and password and click **Log In**.

Step 6 On the top menu bar, choose **SQL Operations** > **SQL Window**.

Step 7 Run the following SQL statement in the SQL window to view the database character set:

show variables like '%character%';

Figure 10-1 SQL execution result



The following is the execution result set of show variables like '%character%'.

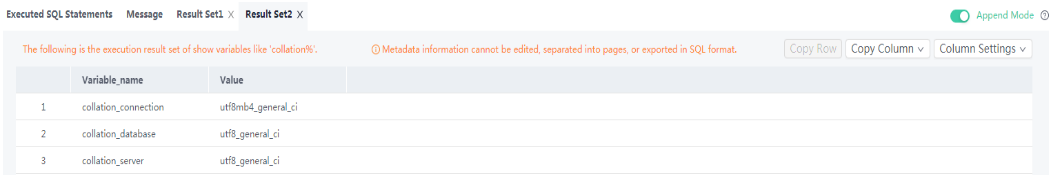
Metadata information cannot be edited, separated into pages, or exported in SQL format.

	Variable_name	Value
1	character_set_client	utf8mb4
2	character_set_connection	utf8mb4
3	character_set_database	utf8
4	character_set_filesystem	binary
5	character_set_results	
6	character_set_server	utf8
7	character_set_system	utf8
8	character_sets_dir	/usr/local/mysql-5.7.27...

Step 8 Run the following SQL statement in the SQL window to view the database coding:

show variables like 'collation%';

Figure 10-2 SQL execution result



The following is the execution result set of show variables like 'collation%'.

Metadata information cannot be edited, separated into pages, or exported in SQL format.

	Variable_name	Value
1	collation_connection	utf8mb4_general_ci
2	collation_database	utf8_general_ci
3	collation_server	utf8_general_ci

Step 9 Change the character set to utf8mb4.

1. Run the following SQL statement to change the database character sets.

ALTER DATABASE *DATABASE_NAME* DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;

2. Run the following SQL statement to change the table character sets.

   ALTER TABLE *TABLE_NAME* DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;

   NOTE

   The SQL statement just changes the character sets of tables. The character sets of fields in the tables are not changed.

3. Run the following SQL statement to change all the field character sets in tables:

   ALTER TABLE *TABLE_NAME* CONVERT TO CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;

NOTE

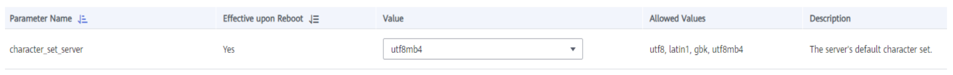
- **character_set_client**, **character_set_connection**, and **character_set_results** are the settings of the client.
- **character_set_system**, **character_set_server**, and **character_set_database** are the settings of the server.
- The priorities of the parameters on the server are as follows: **character_set_database** > **character_set_server** > **character_set_system**.

----End

10.4 How Do I Use the utf8mb4 Character Set to Store Emojis in a GaussDB(for MySQL) Instance?

To store emoji in a GaussDB(for MySQL) instance, ensure that:

- The client outputs the utf8mb4 character set.
- The connection supports the utf8mb4 character set. If you want to use a JDBC connection, download MySQL Connector/J 5.1.13 or a later version and leave **characterEncoding** undefined for the JDBC connection string.
- Configure the instance as follows:
  - Setting **character_set_server** to **utf8mb4**



Parameter Name	Effective upon Reboot	Value	Allowed Values	Description
character_set_server	Yes	utf8mb4	utf8, latin1, gbk, utf8mb4	The server's default character set.

    i. **Log in to the management console.**

    ii. Click in the upper left corner and select a region and a project.

    iii. Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.

    iv. On the **Instances** page, click the instance name.

    v. In the navigation pane on the left, choose **Parameters**. On the **Parameters** tab page, locate **character_set_server** and change its value to **utf8mb4**.

vi. Click **Save**. In the displayed dialog box, click **Yes**.

– Selecting **utf8mb4** for database character set

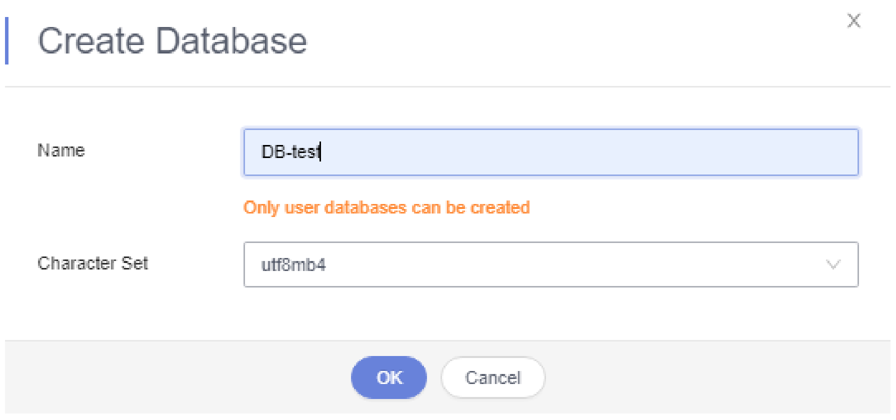
i. On the **Instances** page, locate the instance and click **Log In** in the **Operation** column.

Alternatively, on the **Instances** page, click the instance name to go to the **Basic Information** page. Click **Log In** in the upper right corner of the page.

ii. On the displayed login page, enter the correct username and password and click **Log In**.

iii. On the **Databases** page, click **Create Database**. In the displayed dialog box, enter a database name, select the character set **utf8mb4**, and authorize database permissions for users. Then, click **OK**.

Figure 10-3 Creating a database



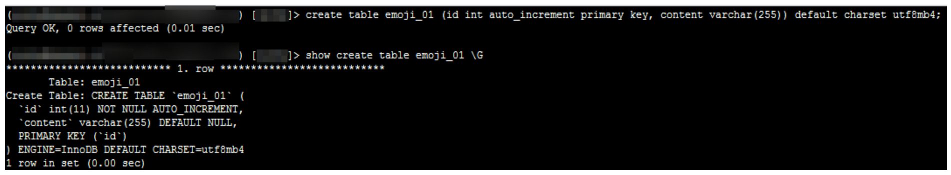
Create Database

Name: DB-test

Only user databases can be created

Character Set: utf8mb4

– Setting the character set of the table to **utf8mb4**



```
([redacted]) [[redacted]]> create table emoji_01 (id int auto_increment primary key, content varchar(255)) default charset utf8mb4;
Query OK, 0 rows affected (0.01 sec)

([redacted]) [[redacted]]> show create table emoji_01 \G
*************************** 1. row ***************************
       Table: emoji_01
Create Table: CREATE TABLE `emoji_01` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `content` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
1 row in set (0.00 sec)
```

FAQs

If you have set **characterEncoding** to **utf8** for the JDBC connection string, or the emoji data cannot be inserted properly after you have performed the above operations, you are advised to set the connection character set to **utf8mb4** as follows:

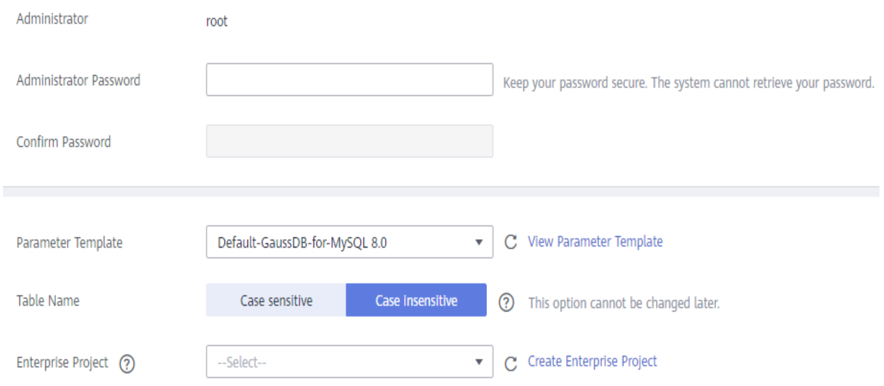
```
String query = "set names utf8mb4";
stat.execute(query);
```

10.5 How Do I Set Case Sensitivity for GaussDB(for MySQL) Table Names?

You can specify case sensitivity for table names when creating an instance on the console or using APIs. It cannot be changed after the instance is created.

- Set **Table Name Case Sensitivity** on the console. For details, see **Buying a DB Instance**.

Figure 10-4 Configuring case sensitivity for table names



Administrator	root	
Administrator Password		Keep your password secure. The system cannot retrieve your password.
Confirm Password		
Parameter Template	Default-GaussDB-for-MySQL 8.0	View Parameter Template
Table Name	Case sensitive / Case insensitive	This option cannot be changed later.
Enterprise Project	--Select--	Create Enterprise Project

- Set **lower_case_table_names** by invoking an API. For details, see **Creating a DB Instance**.

  Value range:

  – **0**: Table names are case sensitive.

  – **1** (default value): Table names are stored in lowercase and are case insensitive.

10.6 Can I Use SQL Commands to Modify Global Parameters of My GaussDB(for MySQL) Instance?

Sorry, you cannot use SQL commands to modify global parameters, but you can modify specific parameters on the console.

Procedure

Step 1 **Log in to the management console.**

Step 2 Click in the upper left corner and select a region and project.

Step 3 Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.

Step 4 On the **Instances** page, click the instance name.

Step 5 In the navigation pane, choose **Parameters**.

Step 6 Change the value of the target parameter and click **Save**.

Step 7 In the displayed dialog box, click **Yes**.

----End

11 Network Security

11.1 How Can I Prevent Untrusted Source IP Addresses from Accessing GaussDB(for MySQL)?

- If you enable public accessibility, your EIP DNS and database port may be vulnerable to hacking. To protect information such as your EIP, DNS, database port, database account, and password, you are advised to specify the range of source IP addresses in the GaussDB(for MySQL) security group to ensure that only trusted source IP addresses can access your instances.
- To prevent your database password from being cracked, specify a strong password and periodically change it.

11.2 How Can I Import the Root Certificate to a Windows or Linux Server?

Importing the Root Certificate to a Windows Server

1. Click **Start** and choose **Run**. In the displayed **Run** dialog box, enter **MMC** and press **Enter**.
2. On the displayed console, choose **File** > **Add/Remove Snap-in**.
3. In the left **Available snap-ins** pane of the displayed dialog box, select **Certificates**. Click **Add** to add the certificate.
4. In the displayed **Certificates snap-in** dialog box, select **Computer account** and click **Next**.
5. In the displayed **Select Computer** dialog box, click **Finish**.
6. In the **Add or Remove Snap-ins** dialog box, click **OK**.
7. On the console, double-click **Certificates**.
8. Right-click **Trusted Root Certification Authorities** and choose **All Tasks** > **Import**.
9. Click **Next**.

10. Click **Browse** to change the file type to **All Files (*.*)**.
11. Locate the downloaded root certificate (a **ca.pem** file) and click **Open**. Then, click **Next**.

NOTICE

You must change the file type to **All Files (*.*)** because **.pem** is not a standard certificate extension name.

12. Click **Next**.
13. Click **Finish**.
14. Click **OK** to complete the import of the root certificate.

Importing the Root Certificate to a Linux Server

You can use a connection tool (such as WinSCP or PuTTY) to upload the certificate to any directory on a Linux Server.

12 Log Management

12.1 Can I Enable general_log for GaussDB(for MySQL)?

No.

If you need to enable general_log for full SQL audit and troubleshooting, you can use **TOP SQL** and **SQL Insights**.

12.2 How Do I View All SQL Statements Executed by GaussDB(for MySQL)?

You can use the visualized database management service Data Admin Service (DAS) to quickly search for target SQL execution records.

Procedure

Step 1 **Log in to the management console.**

Step 2 Click 📍 in the upper left corner and select a region and project.

Step 3 Click ☰ in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.

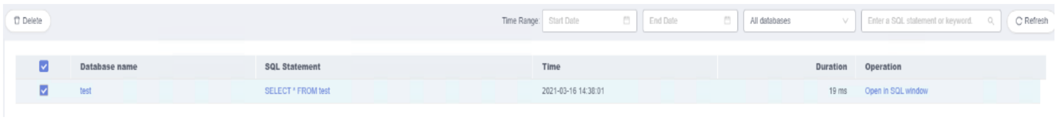
Step 4 On the **Instances** page, locate the instance you want to log in and click **Log In** in the **Operation** column.

Step 5 On the displayed login page, enter the correct username and password and click **Log In**.

Step 6 On the top menu bar, choose **SQL Operations** > **SQL History**.

Step 7 On the displayed page, search for execution information about the target SQL statement by time range, database name, or keyword.

Figure 12-1 SQL history



✓	Database name	SQL Statement	Time	Duration	Operation
✓	test	SELECT * FROM test	2021-03-16 14:38:01	19 ms	Open in SQL window

- To access the **Database Management** page, click a database name.
- To copy and use an SQL statement, click it in the **SQL Statement** column.
- To execute an SQL statement, locate the statement and click **Open in SQL Window** in the **Operation** column.

----End

12.3 How Do I Enable and View Binlog of My GaussDB(for MySQL) Instance?

This section describes how to enable and view binlog, and the impact on GaussDB(for MySQL) performance after binlog is enabled.

- **Enabling Binlog**
- **Viewing Binlog Files**
- **Impact of Enabling Binlog on GaussDB(for MySQL) Performance**

Enabling Binlog

GaussDB(for MySQL) does not support enabling binlog for read replicas. For details about enabling binlog for the primary node, perform the following steps:

Step 1 **Log in to the management console.**

Step 2 Click in the upper left corner and select a region and project.

Step 3 Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.

Step 4 Click the instance name to go to the **Basic Information** page.

Step 5 In the navigation pane, choose **Parameters**.

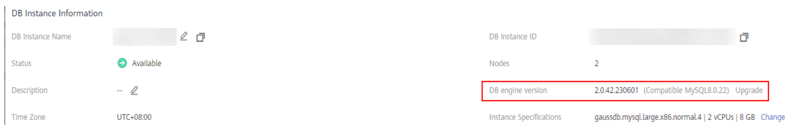
Step 6 Configure parameters as follows:

- If the kernel version is earlier than 2.0.45.230900, search for the **log-bin** parameter, select **ON** from the drop-down list box in the **Value** column, and click **Save**. The modified parameter value is applied only after the DB instance is rebooted.

NOTE

To view the kernel version, click the instance name to go to the **Basic Information** page. In the **DB Instance Information** area, view the **Kernel Version** field.

Figure 12-2 Viewing the kernel version



DB Instance Information

DB Instance Name	[redacted]	DB Instance ID	[redacted]
Status	Available	Nodes	2
Description	--	DB engine version	2.0.42.230601 (Compatible MySQL8.0.22) Upgrade
Time Zone	UTC+08:00	Instance Specifications	gaussdb.mysql.large.x86.normal.4 | 2 vCPUs | 8 GB Change

- If the kernel version is 2.0.45.230900 or later, search for the **rds_global_sql_log_bin** parameter, select **ON** from the drop-down list box in the **Value** column, and click **Save**. The modified parameter value is applied immediately. You do not need to reboot the DB instance.

  After this parameter is changed, connect to the database and run the following command to check whether the binlog is enabled for all threads:

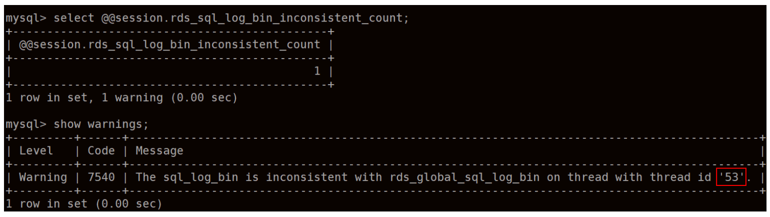
  select @@session.rds_sql_log_bin_inconsistent_count;

  – If the command output is 0, binlog is successfully enabled for all threads, and all statements can be recorded in binlog.

  – If the command out is not 0, run the following command to check the IDs of the threads that binlog is not enabled for:

    show warnings;

    Figure 12-3 Querying the IDs of the threads that binlog is not enabled for

```
mysql> select @@session.rds_sql_log_bin_inconsistent_count;
+----------------------------------------------+
| @@session.rds_sql_log_bin_inconsistent_count |
+----------------------------------------------+
|                                            1 |
+----------------------------------------------+
1 row in set, 1 warning (0.00 sec)

mysql> show warnings;
+---------+------+----------------------------------------------------------------------------------------------+
| Level   | Code | Message                                                                                      |
+---------+------+----------------------------------------------------------------------------------------------+
| Warning | 7540 | The sql_log_bin is inconsistent with rds_global_sql_log_bin on thread with thread id '53'.  |
+---------+------+----------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

    The statements executed in the queried thread IDs may not be recorded in binlog temporarily.

    Check your services based on the obtained thread IDs (for example, **53** in **Figure 12-3**), submit or roll back transactions and execute new transactions (for example, **SELECT 1;**) in a timely manner based on service requirements, or disconnect idle connections and reconnect them.

----End

Viewing Binlog Files

Step 1 Connect to an instance. For details, see **Connect to a DB Instance**.

Step 2 Run the following command to view binlog files:

SHOW BINLOG EVENTS [**IN** '*log_name*'] [FROM *pos*] [LIMIT [*offset*,] *row_count*]**;**

NOTE

If a message indicating that the account permissions are insufficient, use the **root** account.

----End

Impact of Enabling Binlog on GaussDB(for MySQL) Performance

Enabling binlog does not affect SELECT operations, but affects INSERT, UPDATE, DELETE and other write operations.

NOTE

There are no significant differences between GaussDB(for MySQL) binlog and open-source MySQL binlog. The binlog syntax of GaussDB(for MySQL) is fully compatible with that of the open-source MySQL.

12.4 How Do I Change the Binlog Retention Period?

GaussDB(for MySQL) is compatible with the **binlog_expire_logs_seconds** parameter of community version 8.0. You can change binlog retention period using this parameter.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner and select a region and project.

Step 3 Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.

Step 4 On the **Instances** page, click the instance name.

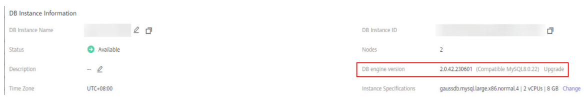
Step 5 In the navigation pane, choose **Parameters**. On the **Parameters** tab, view the following parameters.

- If the kernel version is earlier than 2.0.45.230900, search for the **log-bin** parameter. If the parameter value is **ON**, binlog is enabled.
- If the kernel version is 2.0.45.230900 or later, search for the **rds_global_sql_log_bin** parameter. If the parameter value is **ON**, binlog is enabled.

NOTE

To view the kernel version, click the instance name to go to the **Basic Information** page. In the **DB Instance Information** area, view the **Kernel Version** field.

Figure 12-4 Viewing the kernel version



DB Instance Information

DB Instance Name		DB Instance ID	
Status	Available	Nodes	2
Description	--	DB engine version	2.0.42.230601 (Compatible MySQL8.0.22) Upgrade
Time Zone	UTC+08:00	Instance Specifications	gaussdb.mysql.large.x86.normal.4 | 2 vCPUs | 8 GB Change

Step 6 On the **Parameters** tab, configure **binlog_expire_logs_seconds**.

NOTE

- When a new binlog file is generated, any existing binlog files that have expired will be deleted.
- If no new binlog file is generated, historical binlog files will not be deleted even if they have expired. To delete binlog files manually, connect to the database and run **flush logs;** to forcibly generate a new binlog file.

----End

12.5 How Do I View Deadlock Logs of GaussDB(for MySQL)?

Database deadlock logs are not recorded in error logs. To view deadlock logs, use Data Admin Service (DAS), a visualized and professional database management tool, to quickly execute SQL statements.

Procedure

Step 1 Log in to the management console.

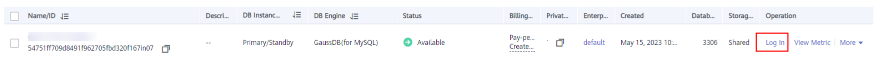
Step 2 Click in the upper left corner and select a region and a project.

Step 3 Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.

Step 4 On the **Instances** page, locate the instance you want to log in and click **Log In** in the **Operation** column.

Step 5 On the displayed login page, enter the correct username and password and click **Log In**.

Figure 12-5 Logging in to an instance



Name/ID	Descri...	DB Instanc...	DB Engine	Status	Billing...	Privat...	Enterp...	Created	Datab...	Storag...	Operation
[redacted] 54751ff709d8491f962705fbd320f167in07	--	Primary/Standby	GaussDB(for MySQL)	Available	Pay-pe... Create...	·	default	May 15, 2023 10:...	3306	Shared	Log In View Metric More

Step 6 Select the target database and click **SQL Operations** > **SQL Window**.

Step 7 In the displayed SQL window, run **show engine innodb status** to view the latest deadlock logs of the selected database. Use the keyword **LATEST DETECTED DEADLOCK** to locate the latest deadlock logs. The latest deadlock logs will overwrite the historical deadlock logs.

----End

13 Version Upgrade

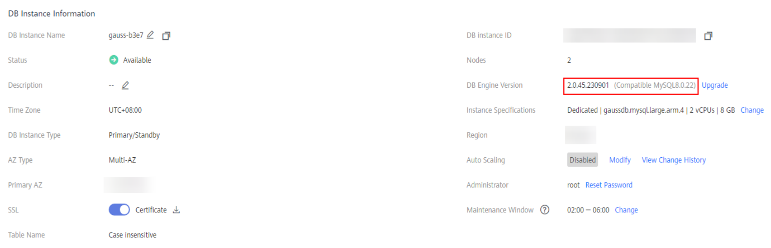
13.1 How Can I Check the Version of a GaussDB(for MySQL) Instance?

This section describes how to check the version of a GaussDB(for MySQL) instance.

Method 1: Checking the Version on the GaussDB(for MySQL) Console

1. **Log in to the management console.**
2. Click in the upper left corner and select a region and project.
3. Click in the upper left corner of the page, choose **Databases** > **GaussDB(for MySQL)**.
4. On the **Instances** page, click the instance name.
5. On the **Basic Information** page, check the instance version.

Figure 13-1 Checking the instance version on the GaussDB(for MySQL) console



DB Instance Information

DB Instance Name	gauss-b3e7	DB instance ID	
Status	Available	Nodes	2
Description	--	DB Engine Version	2.0.45.230901 (Compatible MySQL8.0.22) Upgrade
Time Zone	UTC+08:00	Instance Specifications	Dedicated | gaussdb.mysql.large.arm.4 | 2 vCPUs | 8 GB Change
DB Instance Type	Primary/Standby	Region	
AZ Type	Multi-AZ	Auto Scaling	Disabled Modify View Change History
Primary AZ		Administrator	root Reset Password
SSL	Certificate	Maintenance Window	02:00 – 06:00 Change
Table Name	Case insensitive		

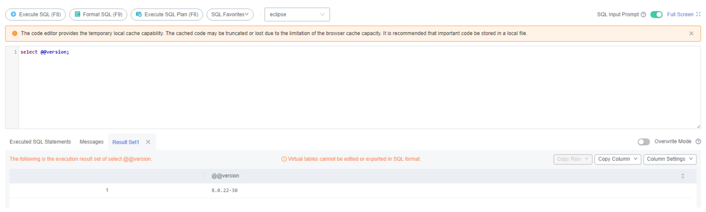
Method 2: Checking the Instance Version on the DAS Console

1. Log in to the target DB instance.
2. On the top menu bar, choose **SQL Operations** > **SQL Query**.

3. Run the following statement to check the instance version:

   select @@version;

   Figure 13-2 Checking the instance version on the DAS console

   The code editor provides the temporary local cache capability. The cached code may be truncated or lost due to the limitation of the browser cache capacity. It is recommended that important code be stored in a local file.

   ```
   select @@version;
   ```

   The following is the execution result set of select @@version.

   Virtual tables cannot be edited or exported in SQL format.

	@@version
1	8.0.22-30