TaurusDB

FAQs

Issue 02

Date 2025-07-22

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: https://www.huaweicloud.com/intl/en-us/

Contents

1 Product Consulting

1.1 What Should I Pay Attention to When Using TaurusDB?

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 TaurusDB 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. Precautions after TaurusDB instances are created:

   After your instance is created, you do not need to perform basic database O&M operations, such as applying HA and security patches, but 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 Response of Websites When They Use TaurusDB?

To solve this problem:

- Check the performance of TaurusDB instances on the TaurusDB console.

- Compare the database connection status of local databases and TaurusDB instances. This problem depends on web applications.

1.3 Does TaurusDB Support Automatic Failover?

Yes. During the creation of a TaurusDB instance, a primary node and a read replica are both created. If the primary node fails, the read replica is automatically promoted to the primary node to provide services, and the original primary node is demoted to be a read replica.

1.4 Does TaurusDB Support Decoupling of Compute and Storage?

TaurusDB supports the decoupling of compute and storage, improving high availability and the experience in backup and restoration, upgrade and capacity expansion.

2 Database Connections

2.1 Can an External Server Access TaurusDB?

DB Instance Associated with an EIP

For a TaurusDB instance that has been associated with an EIP, you can access it through the EIP.

For details, see:

Connecting to a DB Instance over a Public Network

DB Instance Not Associated with an EIP

- Enable a VPN in a VPC and use the VPN to connect to the TaurusDB instance.
- Create a TaurusDB instance and an ECS in the same VPC and access TaurusDB through the ECS.

For details, see:

Connecting to a DB Instance over a Private Network

2.2 What Do I Do If the Number of TaurusDB 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 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 Parameters in a Parameter Template**.
3. Optimize slow queries.

2.3 What Is the Maximum Number of Connections to a TaurusDB Instance?

TaurusDB 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 TaurusDB instance.

Changing the Maximum Number of Connections

The number of connections can be changed online. For details, see **Modifying Parameters in 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 occupied by 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 2-1 Default values of 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

2.4 What Should I Do If an ECS Cannot Connect to a TaurusDB Instance?

Perform the following steps to identify the problem:

Step 1 Check whether the ECS and TaurusDB instance are located 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 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.
- If no security group has been created, go to the VPC console from the ECS details page and create a security group.

Step 3 Check whether the ECS can connect to the instance over the instance port.

The default port of a cluster TaurusDB instance is **3306**.

```
telnet <IP address> {Port number}
```

- 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

2.5 How Can I Connect to a MySQL Database Through JDBC?

Although the SSL certificate is optional if you choose to connect to a database through Java database connectivity (JDBC), you are advised to download the SSL certificate to encrypt the connections for security purposes. By default, SSL data encryption is enabled for newly created TaurusDB instances. Enabling SSL will increase the network connection response time and CPU usage. Before enabling SSL, evaluate the impact on service performance.

Prerequisites

Familiarize yourself with:

- Computer basics
- Java programming language
- JDBC knowledge

Connection with the SSL Certificate

The SSL certificate needs to be downloaded and verified for connecting to databases.

NOTE

If the **ssl_type** value of a database user is **x509**, this method is unavailable.

To check the **ssl_type** value of the current user, run the following command:

select ssl_type from mysql.user where user = '*xxx***';**

Step 1 Download the CA certificate or certificate bundle.

1. On the **Instances** page, click the instance name to go to the **Basic Information** page.
2. In the **DB Instance Information** area, click ⤓ next to **SSL**.

Step 2 Use keytool to generate a truststore file using the CA certificate.

```
<keytool installation path> ./keytool.exe -importcert -alias <MySQLCACert> -file <ca.pem> -keystore
<truststore_file> -storepass <password>
```

Table 2-2 Parameter description

Parameter	Description
<keytool installation path>	Bin directory in the JDK or JRE installation path, for example, **C:\Program Files (x86)\Java\jdk11.0.7\bin**.

Parameter	Description
<MySQLCACert>	Name of the truststore file. Set it to a name specific to the service for future identification.
<ca.pem>	Name of the CA certificate downloaded and decompressed in **Step 1**, for example, **ca.pem**.
<truststore_file>	Path for storing the truststore file.
<password>	Password of the truststore file.

Code example (using keytool in the JDK installation path to generate the truststore file):

```
Owner:  CN=MySQL_Server_8.0.22_Auto_Generated_CA_Certificate
Issuer: CN=MySQL_Server_8.0.22_Auto_Generated_CA_Certificate
Serial number: 1
Valid from: Thu Feb 16 11:42:43 EST 2017 until: Sun Feb 14 11:42:43 EST 2027
Certificate fingerprints:
    MD5: 18:87:97:37:EA:CB:0B:5A:24:AB:27:76:45:A4:78:C1
    SHA1: 2B:0D:D9:69:2C:99:BF:1E:2A:25:4E:8D:2D:38:B8:70:66:47:FA:ED
    SHA256:C3:29:67:1B:E5:37:06:F7:A9:93:DF:C7:B3:27:5E:09:C7:FD:EE:2D:18:86:F4:9C:40:D8:26:CB:DA:95:
A0:24
    Signature algorithm name: SHA256withRSA Subject Public Key Algorithm: 2048-bit RSA key
    Version: 1
    Trust this certificate? [no]: y
    Certificate was added to keystore
```

Step 3 Connect to the MySQL instance through JDBC.

```
jdbc:mysql://<instance_ip>:<instance_port>/<database_name>?
requireSSL=<value1>&useSSL=<value2>&verifyServerCertificate=<value3>&trustCertificateKeyStoreUrl=f
ile:
<truststore_file>&trustCertificateKeyStorePassword=<password>
```

Table 2-3 Parameter description

Parameter	Description
<instance_ip>	IP address of the DB instance. **NOTE** • If you are accessing the instance through an ECS, *<instance_ip>* is the private IP address of the instance. You can view the private IP address in the **Network Information** area on the **Basic Information** page. • If you are accessing the instance through a public network, *<instance_ip>* is the EIP that has been bound to the instance. You can view the EIP in the **Network Information** area on the **Basic Information** page.
<instance_port>	Database port of the instance. The default port is **3306**. **NOTE** You can view the database port in the **Network Information** area on the **Basic Information** page.
<database_name>	Database name used for connecting to the instance. The default value is **mysql**.

Parameter	Description
<value1>	Value of **requireSSL**, indicating whether the server supports SSL. It can be either of the following: ● **true**: The server supports SSL. ● **false**: The server does not support SSL. **NOTE** For details about the relationship between **requireSSL** and **sslmode**, see **Table 2-4**.
<value2>	Value of **useSSL**, indicating whether the client uses SSL to connect to the server. It can be either of the following: ● **true**: The client uses SSL to connect to the server. ● **false**: The client does not use SSL to connect to the server. **NOTE** For details about the relationship between **useSSL** and **sslmode**, see **Table 2-4**.
<value3>	Value of **verifyServerCertificate**, indicating whether the client verifies the server certificate. It can be either of the following: ● **true**: The client verifies the server certificate. ● **false**: The client does not verify the server certificate. **NOTE** For details about the relationship between **verifyServerCertificate** and **sslmode**, see **Table 2-4**.
<truststore_file>	Path for storing the truststore file configured in **Step 2**.
<password>	Password of the truststore file configured in **Step 2**.

Table 2-4 Relationship between connection parameters and sslmode

useSSL	requireSSL	verifyServerCertificate	sslMode
false	N/A	N/A	DISABLED
true	false	false	PREFERRED
true	true	false	REQUIRED
true	N/A	true	VERIFY_CA

Code example (Java code for connecting to a MySQL instance):

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.sql.SQLException;
```

```
public class JDBCTest {
   //There will be security risks if the username and password used for authentication are directly written into code. Store the username and password in ciphertext in the configuration file or environment variables.
   //In this example, the username and password are stored in the environment variables. Before running the code, set environment variables EXAMPLE_USERNAME_ENV and EXAMPLE_PASSWORD_ENV as needed.
    static final String USER = System.getenv("EXAMPLE_USERNAME_ENV");
    static final String PASS = System.getenv("EXAMPLE_PASSWORD_ENV");

    public static void main(String[] args) {
        Connection conn = null;
        Statement stmt = null;

        String url = "jdbc:mysql://<instance_ip>:<instance_port>/<database_name>?requireSSL=true&useSSL=true&verifyServerCertificate=true&trustCertificateKeyStoreUrl=file:<truststore_file>&trustCertificateKeyStorePassword=<password>";

        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            conn = DriverManager.getConnection(url, USER, PASS);

            stmt = conn.createStatement();
            String sql = "show status like 'ssl%'";
            ResultSet rs = stmt.executeQuery(sql);

            int columns = rs.getMetaData().getColumnCount();
            for (int i = 1; i <= columns; i++) {
                System.out.print(rs.getMetaData().getColumnName(i));
                System.out.print("\t");
            }

            while (rs.next()) {
                System.out.println();
                for (int i = 1; i <= columns; i++) {
                    System.out.print(rs.getObject(i));
                    System.out.print("\t");
                }
            }
            rs.close();
            stmt.close();
            conn.close();
        } catch (SQLException se) {
            se.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            // release resource ....
        }
    }
}
```

----End

Connection Without the SSL Certificate

NOTE

You do not need to download the SSL certificate because certificate verification on the server is not required.

Step 1 Connect to your TaurusDB instance through JDBC.

```
jdbc:mysql://<instance_ip>:<instance_port>/<database_name>?useSSL=false
```

Table 2-5 Parameter description

Parameter	Description
<instance_ip>	IP address of the instance. **NOTE** ● If you are accessing the instance through an ECS, *<instance_ip>* is the private IP address of the instance. You can view the private IP address in the **Network Information** area on the **Basic Information** page. ● If you are accessing the instance through a public network, *<instance_ip>* is the EIP that has been bound to the instance. You can view the EIP in the **Network Information** area on the **Basic Information** page.
<instance_port>	Database port of the instance. The default port is **3306**. **NOTE** You can view the database port in the **Network Information** area on the **Basic Information** page.
<database_name>	Database name used for connecting to the instance. The default value is **mysql**.

Code example (Java code for connecting to a MySQL instance):

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class MyConnTest {
    final public static void main(String[] args) {
        Connection conn = null;
          // set sslmode here.
        // no ssl certificate, so do not specify path.
        String url = "jdbc:mysql://192.168.0.225:3306/my_db_test?useSSL=false";
        try {
            Class.forName("com.mysql.jdbc.Driver");
                        //There will be security risks if the username and password used for authentication are directly written into code. Store the username and password in ciphertext in the configuration file or environment variables.
                        //In this example, the username and password are stored in the environment variables. Before running the code, set environment variables EXAMPLE_USERNAME_ENV and EXAMPLE_PASSWORD_ENV as needed.
                        conn = DriverManager.getConnection(url, System.getenv("EXAMPLE_USERNAME_ENV"), System.getenv("EXAMPLE_PASSWORD_ENV"));
            System.out.println("Database connected");

            Statement stmt = conn.createStatement();
            ResultSet rs = stmt.executeQuery("SELECT * FROM mytable WHERE columnfoo = 500");
            while (rs.next()) {
                System.out.println(rs.getString(1));
            }
            rs.close();
            stmt.close();
            conn.close();
        } catch (Exception e) {
            e.printStackTrace();
            System.out.println("Test failed");
        } finally {
            // release resource ....
        }
```

```
   }
}
```

----End

Related Issues

- Symptom

  When you use JDK 8.0 or a later version to connect to your instance with an SSL certificate downloaded, an error similar to the following is reported:

  ```
  javax.net.ssl.SSLHandshakeException: No appropriate protocol (protocol is disabled or
  cipher suites are inappropriate)
      at sun.security.ssl.HandshakeContext.<init>(HandshakeContext.java:171) ~[na:1.8.0_292]
      at sun.security.ssl.ClientHandshakeContext.<init>(ClientHandshakeContext.java:98) ~
  [na:1.8.0_292]
      at sun.security.ssl.TransportContext.kickstart(TransportContext.java:220) ~
  [na:1.8.0_292]
      at sun.security.ssl.SSLSocketImpl.startHandshake(SSLSocketImpl.java:428) ~
  [na:1.8.0_292]
      at
  com.mysql.cj.protocol.ExportControlled.performTlsHandshake(ExportControlled.java:316) ~
  [mysql-connector-java-8.0.17.jar:8.0.17]
      at
  com.mysql.cj.protocol.StandardSocketFactory.performTlsHandshake(StandardSocketFactory.java
  :188) ~[mysql-connector-java8.0.17.jar:8.0.17]
      at
  com.mysql.cj.protocol.a.NativeSocketConnection.performTlsHandshake(NativeSocketConnection.
  java:99) ~[mysql-connector-java8.0.17.jar:8.0.17]
      at
  com.mysql.cj.protocol.a.NativeProtocol.negotiateSSLConnection(NativeProtocol.java:331) ~
  [mysql-connector-java8.0.17.jar:8.0.17]
  ... 68 common frames omitted
  ```

- Solution

  Specify the corresponding parameter values in the code link of **Step 3** based on the JAR package used by the client. Example:

  - mysql-connector-java-5.1.*xx*.jar

    ```
    jdbc:mysql://<instance_ip>:<instance_port>/<database_name>?

    requireSSL=true&useSSL=true&verifyServerCertificate=true&trustCertificateKeyStoreUrl=file:

     <truststore_file>&trustCertificateKeyStorePassword=<password>&
    enabledTLSProtocols=TLSv1.2
    ```

  - mysql-connector-java-8.0.*xx*.jar

    ```
    jdbc:mysql://<instance_ip>:<instance_port>/<database_name>?

    requireSSL=true&useSSL=true&verifyServerCertificate=true&trustCertificateKeyStoreUrl=file:
     <truststore_file>&trustCertificateKeyStorePassword=<password>& tlsVersions =TLSv1.2
    ```

2.6 How Can I Create and Connect to an ECS?

1. For details about how to create an ECS, see *Elastic Cloud Server User Guide*.
   - The ECS to be created must be in the same VPC with the TaurusDB instance to which it connects.
   - Configure a security group to allow the ECS to access the TaurusDB instance through the IP address.
2. For details on how to connect to the ECS, see section "Logging In to an ECS" in the *Elastic Cloud Server User Guide*.

2.7 What Should I Do If a Database Client Problem Causes a Connection Failure?

Check the following items to troubleshoot TaurusDB connection failures caused by a client problem:

1. ECS security policy

   In Windows, check whether the TaurusDB instance port is enabled in the Windows security policy. In Linux, run **iptables** to check whether the instance port is enabled in firewall settings.

2. Application configuration

   Check whether the connection address, port parameter configuration, and JDBC connection parameter configuration are correct.

3. Username or 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)

NOTE

If the problem persists, contact post-sales technical support.

2.8 Why Cannot I Ping My EIP After It Is Bound to a DB Instance?

Fault Locating

1. Check security group rules.
2. Check network ACLs.
3. Ping the ECS to the instance in the same region.

Solution

1. Check security group rules.

   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 **Database** > **TaurusDB**.

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

e. In the **Network Information** area on the **Basic Information** page, click the security group.

f. Check whether the ECS NIC security group allows the inbound ICMP traffic.

Table 2-6 Security group rules

Direction	Type	Protocol/Port Range	Source IP Address
Inbound	IPv4	Any: Any	0.0.0.0/0 (all IP addresses)
Inbound	IPv4	ICMP: Any	0.0.0.0/0 (all IP addresses)

2. Check network ACLs.

   a. Check the network ACL status.

   b. Check whether the NIC to which the EIP bound belongs to the subnet associated with the network ACL.

   c. If the network ACL is enabled, add an ICMP rule to allow traffic.

   NOTE

   The default network ACL rule denies all incoming and outgoing packets. After the network ACL is disabled, the default rule still takes effect.

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

   If the affected EIP can be pinged from another ECS in the same region, the virtual network is functional. In such a case, contact customer service for technical support.

2.9 What Can I Do If the Connection Test Failed?

Fault Locating

1. Check security group rules.
2. Check network ACLs.
3. Check the NIC information of ECSs.
4. Check the disconnected ports.

Solution

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 **Database** > **TaurusDB**.

Step 4 On the **Instances** page, click the instance name to go to the **Basic Information** page. In the **Network Information** area of the page, view the VPC where the TaurusDB instance is located.

Step 5 Check whether the TaurusDB instance to which distributed transactions are added is in the same VPC as an ECS.

- If they are in the same VPC, see **Why Does Communication Fail Between Two ECSs in the Same VPC or Packet Loss Occur When They Communicate?**
- If they are in different VPCs:
  - Public access: Bind an EIP to the instance. For details, see **Binding an EIP**.
  - Private access: Create a VPC peering connection.
  - Change the VPC hosting the ECS to the same as that hosting the TaurusDB. For details, see **Changing a VPC**.

----End

2.10 Can I Access a TaurusDB Instance over an Intranet Connection Across Regions?

By default, 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.
- 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.

2.11 Are There Any Potential Risks If There Are Too Many Connections to a TaurusDB Instance?

If there is an excessive number of TaurusDB 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.

2.12 What Should I Do If an ECS and a TaurusDB Instance Deployed in Different VPCs Cannot Communicate with Each Other?

When a TaurusDB instance and an ECS are deployed in different VPCs of the same region, they cannot communicate with each other through a private network. After a TaurusDB instance is created, you cannot change its VPC.

Solution

1. Create a VPC peer connection.
2. Change the VPC hosting the ECS to the same as that hosting the TaurusDB. For details, see **Changing a VPC**.

2.13 How Do I View All IP Addresses Connected to a Database?

You can run the following SQL statement on the database to query the number of connected IP addresses:

```
SELECT substring_index(host, ':',1) AS host_name,state,count(*) FROM
information_schema.processlist GROUP BY state,host_name;
```

3 Client Installation

3.1 How Can I Install the MySQL Client?

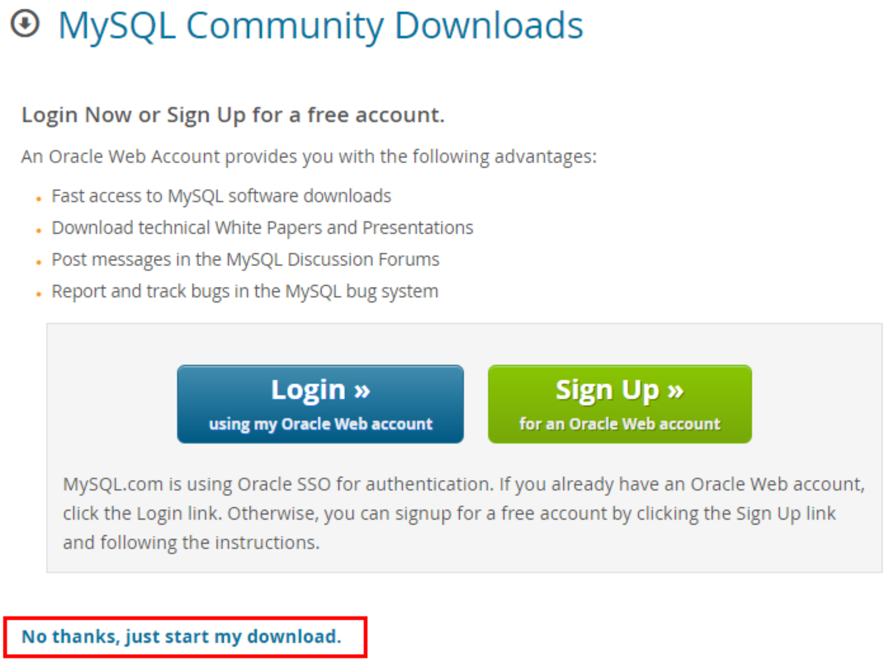
MySQL provides client installation packages for different OSs on its official website. Download the **MySQL 8.0 client installation package** or **packages of other versions**. The following uses Red Hat Linux as an example to show how to obtain the required installation package and install it.

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 3-1 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.

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 3-1 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 of the TaurusDB database administrator account. The default username 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

4 Database Migration

4.1 What Types of DB Engines Does TaurusDB 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 TaurusDB.

  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 TaurusDB.

5 Database Permissions

5.1 Does TaurusDB Provide the Root Account or Super Permission?

TaurusDB 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 TaurusDB. This is a major difference between cloud databases and on-premises MySQL databases. To ensure stable running of instances, TaurusDB does not provide the super permission for the **root** user.

If you need to perform operations that require super permissions, TaurusDB provides alternative methods.

For example:

You can modify parameter values only on the TaurusDB 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 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 TaurusDB Using mysqldump**.

6 Database Performance

6.1 What Should I Do If the CPU Usage of My TaurusDB Instance Is High?

If the CPU usage is high or close to 100% when you use TaurusDB, 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 TaurusDB logs, see **Viewing Slow Query Logs**.
2. View the CPU usage of your TaurusDB instance to facilitate problem locating.
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.

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

Scenario

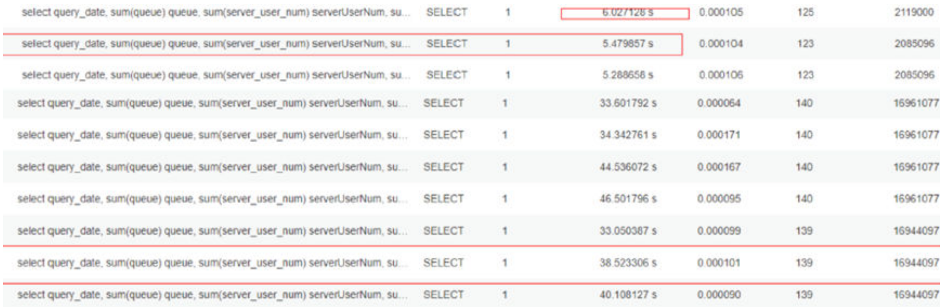
On your TaurusDB instance, an SQL query that ran at 11:00 and was expected to take 8 seconds took more than 30 seconds.

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.

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.

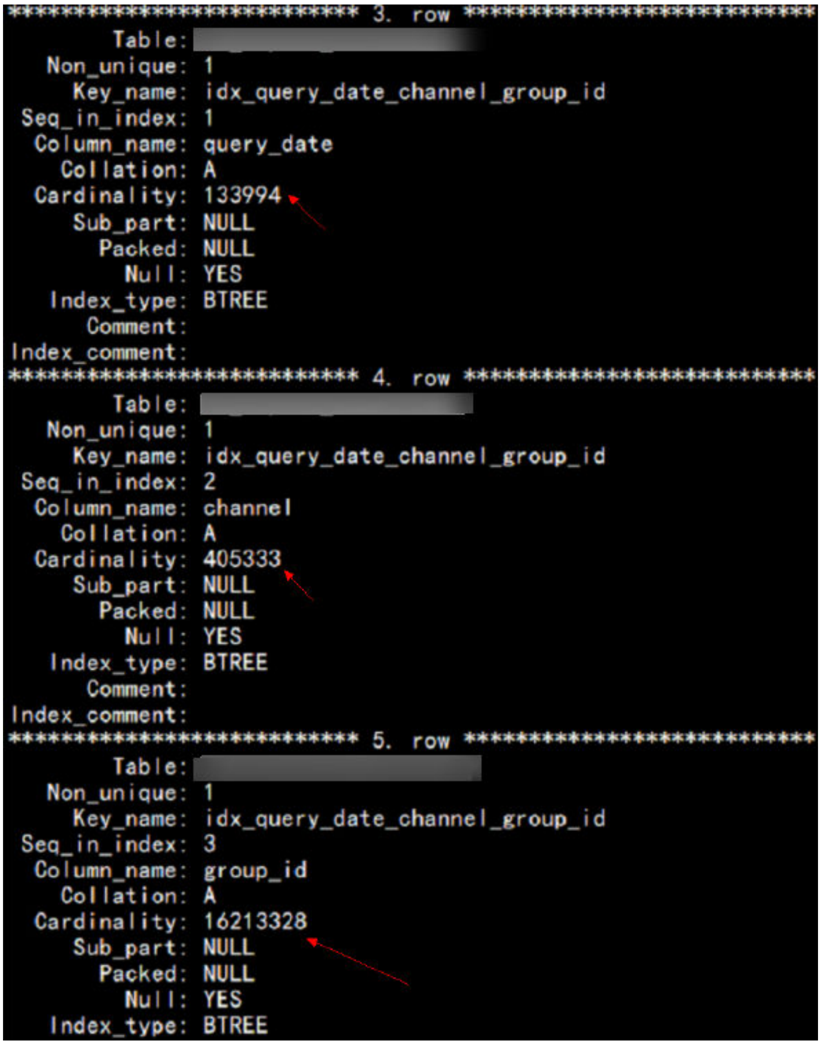
Figure 6-1 Slow query logs



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 6-2 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

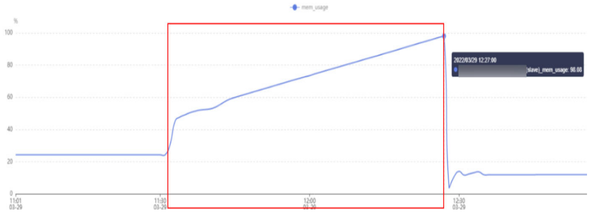
1. Check whether the slow query was caused by insufficient CPU resources.
2. Check whether the table structure is properly designed and whether index settings are correct.
3. Execute the **ANALYZE TABLE** statement periodically to prevent incorrect execution plans because performing a large number of INSERT or DELETE operations for table data may result in outdated statistics.

6.3 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 TaurusDB instance kept increasing from 11:30 to 12:27 and reached the memory limit.

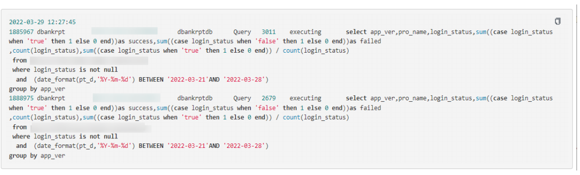
Figure 6-3 Memory usage



Possible Causes

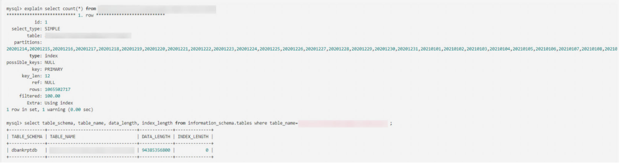
1. Check the **processlist.log** file. In this example, shown below, there were two slow SQL statements in that time period.

Figure 6-4 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.

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

Scenario

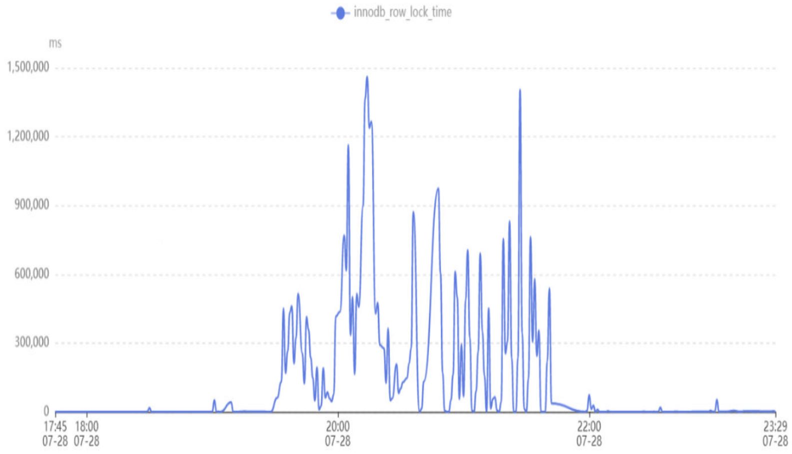
Error code 1205 was reported:

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

Possible Causes

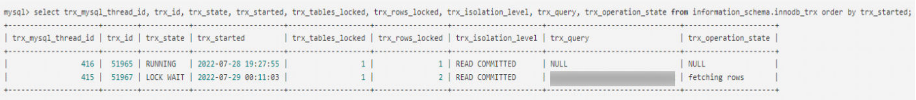
1. Check the value of the monitoring metric **Row Lock Time**. In this example, the value of this metric was high, so we know there were lock conflicts in the system.

   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.

6.5 What Is the CPU Usage of a TaurusDB Instance with Empty Load?

A TaurusDB 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.

7 Database Usage

7.1 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.

7.2 How Do I Add Columns Using INSTANT?

TaurusDB is compatible with open-source MySQL 8.0.22, so you can use **ALGORITHM=INSTANT** to quickly add columns, preventing lock waiting from affecting workloads 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 **Database** > **TaurusDB**.

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

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

You can use LOAD DATA to import local data to TaurusDB.

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 TaurusDB 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 TaurusDB 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.

7.4 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:

1. 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.

2. 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');
   ```

3. 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.

4. When inserting data with primary keys, try to insert them in a sequential order of the primary keys. You can use AUTO_INCREMENT.

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

   Example:

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

   Inserting data in a sequential order of primary keys: 1 2 4 6 8

5. 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.

6. Avoid modifying primary keys during service operations.

   Modifying primary keys requires modifying the index structure, which can be costly.

7. Reduce the length of primary keys as much as possible if the business permits.

8. Do not use foreign keys to maintain foreign key relationships. Use programs instead.

9. 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:

1. Keep the index field as small as possible.
2. Select a column with high distinction as the index column.
3. 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.
4. To create an index, insert data first and then run the **alter table add index** command.

7.5 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.

8 Backups

8.1 How Long Can a TaurusDB Instance Backup Be Retained?

Automated backup data is kept based on the backup retention period you specified. For details, see **Configuring an Automated 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 database storage space.

8.2 How Do I Clear TaurusDB Backup Space?

The TaurusDB backup space stores automated backups and manual backups.

- **Automated full and incremental backups**

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

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

8.3 How Can I Back Up a TaurusDB 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.

You are advised to store the data to OBS for higher data reliability and service assurance.

8.4 How Do I View My Backup Storage Usage?

In the **Storage/Backup Space** area of the **Basic Information** page on the console, you can view the backup space usage.

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 **Database** > **TaurusDB**.

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

Step 5 On the **Basic Information** page, view the backup space usage in the **Backup Space** area.

NOTE

There are full and incremental backups of an instance in the backup space.

Free backup storage space of the same size as your purchased storage space are provided.

----End

8.5 Why Has My Automated Backup Failed?

Automated backups may fail for the following reasons:

1. The network environment is unstable due to issues such as network delay or interruption. TaurusDB will detect these problems and trigger another automated backup half an hour later. You can also perform a manual backup before then.
2. Multi-task executions are complicated, resulting in problems such as task waiting or interruptions. TaurusDB will detect these problems and trigger another automated backup half an hour later. You can also perform a manual backup.
3. An instance is unavailable, possibly because the instance is faulty or being modified. TaurusDB will trigger an automated backup when the instance status becomes available. You can also perform a manual backup before then.
4. A parameter is changed incorrectly. For example, an instance may become faulty after a parameter template containing incorrect parameters was applied to it. You can check whether original and current values are correct, check whether any related parameters also need to be changed, reset the parameter template, or reboot the instance.
5. An error occurred during data import.

   If system catalog records get lost due to improper data import, you can use DRS to import the data again.

6. If the issue persists, contact technical support.

8.6 How Is TaurusDB Backup Data Billed?

All the TaurusDB backups are stored on OBS without occupying the storage of your DB instances. TaurusDB provides free backup space of the same size as your DB instance storage space.

The lifecycle of automated backups is the same as that of the DB instance. If you delete a DB instance, its automated backups are also deleted, but manual backups will not be automatically deleted.

For example, if you create 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.

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.

9 Database Parameter Modification

9.1 How Can I Change the Time Zone?

TaurusDB 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 **Database** > **TaurusDB**.

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**.

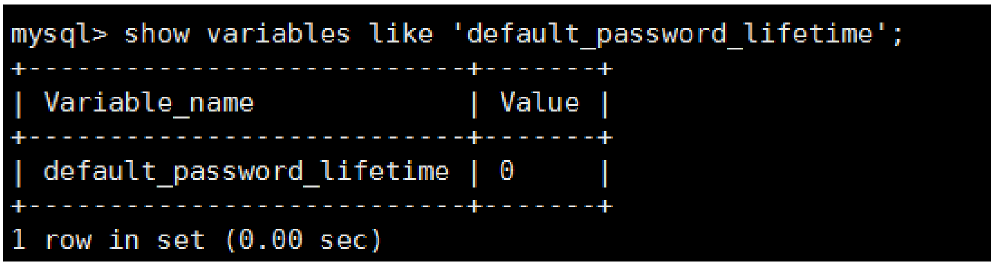
Step 8 In the displayed dialog box, click **OK**.

----End

9.2 How Do I Configure a Password Expiration Policy for TaurusDB Instances?

In TaurusDB kernel version 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

- Change the value of the **default_password_lifetime** parameter on the TaurusDB console.

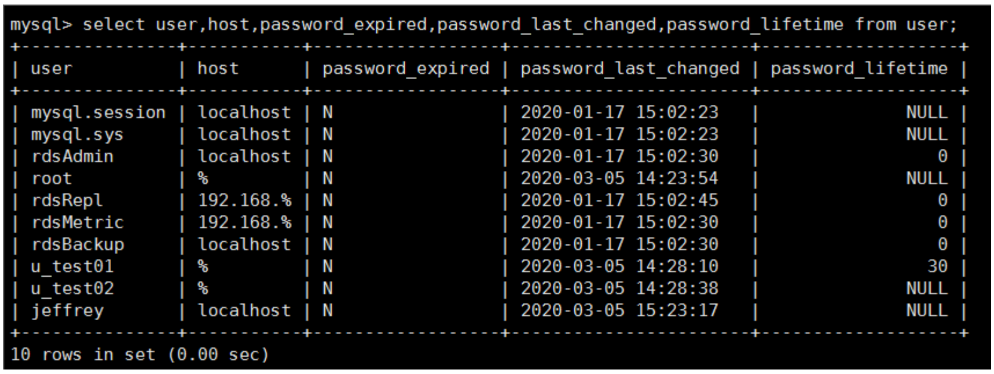
  For details, see **Modifying Parameters in a Parameter Template**.
- Run the following command to change the value of **default_password_lifetime**:

  mysql> set global default_password_lifetime=0;

Checking the Password Expiration Date of All Users

Run the following command:

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

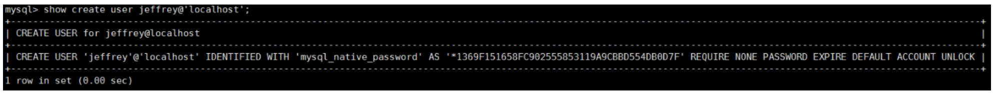


```
mysql> select user,host,password_expired,password_last_changed,password_lifetime from user;
+---------------+-----------+------------------+-----------------------+-------------------+
| user          | host      | password_expired | password_last_changed | password_lifetime |
+---------------+-----------+------------------+-----------------------+-------------------+
| 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:

mysql> 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

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

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

- Setting the password to follow the global expiration policy

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

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

9.3 How Do I Ensure that the Database Character Set of a TaurusDB Instance Is Correct?

UTF-8 supports 4 byte characters, but TaurusDB 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.

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 **Database** > **TaurusDB**.

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 in the SQL window to view the database character set:

show variables like '%character%';

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

show variables like 'collation%';

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

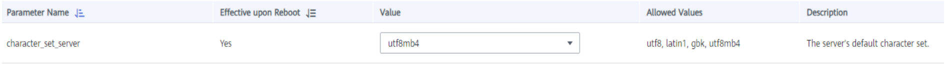
- **character_set_client**, **character_set_connection**, and **character_set_results** are the settings of the client.
- **character_set_system**, **character_set_server**, and **character_set_database** are the settings of the server.
- The priorities of the parameters on the server are as follows: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

----End

9.4 How Do I Use the utf8mb4 Character Set to Store Emojis in a TaurusDB Instance?

To store emoji in a TaurusDB instance, ensure that:

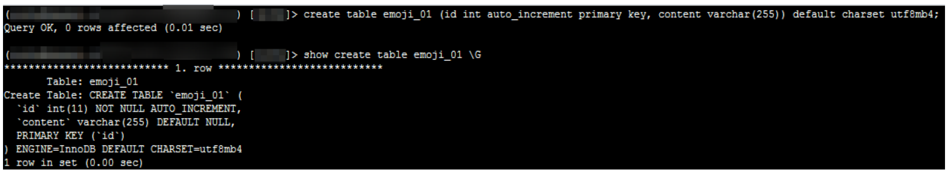
- The client outputs the utf8mb4 character set.
- The connection supports the utf8mb4 character set. If you want to use a JDBC connection, download MySQL Connector/J 5.1.13 or a later version and leave **characterEncoding** undefined for the JDBC connection string.
- Configure the TaurusDB instance as 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 project.
    iii. Click ☰ in the upper left corner of the page, choose **Database** > **TaurusDB**.
    iv. On the **Instances** page, click the instance name.
    v. In the navigation pane, 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**.

- 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);
```

9.5 How Do I Set Case Sensitivity for TaurusDB 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.
- Set **lower_case_table_names** by invoking an API.

  Value range:

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

9.6 Can I Use SQL Commands to Modify Global Parameters?

Sorry, you cannot use SQL commands to modify global parameters, but you can modify specific parameters on the TaurusDB 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 **Database** > **TaurusDB**.

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 **OK**.

----End

10 Network Security

10.1 What Security Protection Policies Does TaurusDB Have?

Network

- TaurusDB allows you to run your instances in a VPC, ensuring that the DB instances are isolated from other services.
- TaurusDB uses security groups to ensure that only trusted sources can access your instances.
- TaurusDB supports SSL connections to encrypt data during transmission.

Management

You can use Identity and Access Management (IAM) to manage TaurusDB permissions.

10.2 How Can I Prevent Untrusted Source IP Addresses from Accessing TaurusDB?

- 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 set the range of source IP addresses in the TaurusDB security group to ensure that only trusted source IP addresses can access your DB instances.
- To prevent your database password from being cracked, set a strong password and periodically change it.

10.3 How Do I Configure a Security Group to Enable Access to TaurusDB Instances?

- When you attempt to connect to a TaurusDB instance through a private network, check whether the ECS and TaurusDB instance are in the same security group.
  - If yes, they can communicate with each other by default. No security group rule needs to be configured.
  - If no, you need to configure security group rules for them, separately.
    - TaurusDB instance: Configure an **inbound rule** for the security group associated with the TaurusDB instance.
    - ECS: The default security group rule allows all outbound data packets. In this case, you do not need to configure a security group rule for the ECS. If not all outbound traffic is allowed in the security group, you may need to configure an outbound rule for the ECS to allow all outbound packets.
- When you attempt to connect to a TaurusDB instance through an EIP, configure an **inbound rule** for the security group associated with the TaurusDB instance.

10.4 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.

10.5 How Do I Manage and Ensure TaurusDB Security?

For security reasons, do not use the username and password of your account. We recommend that you create IAM users and other users (if necessary) for your database users.

An error may occur if you do not have sufficient permissions or your account configuration is incorrect. For example, you fail to create instances if you do not have the permissions to do so. If required, contact your IAM administrator to assign the permissions

11 Log Management

11.1 Can I Enable general_log for TaurusDB?

No.

11.2 How Do I View All SQL Logs Executed by TaurusDB?

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

Querying SQL Logs Through DAS

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 **Database** > **TaurusDB**.

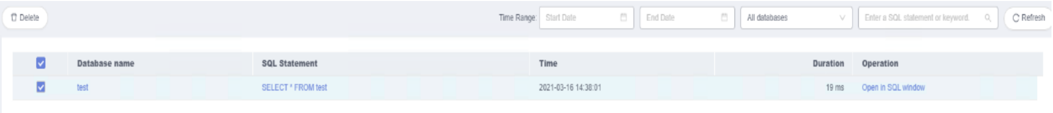
Step 4 On the **Instances** page, locate the instance you want to log in and click **Log In** in the **Operation** column.

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 11-1 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 a SQL statement, click it in the **SQL Statement** column.
- To execute a SQL statement, locate the statement and click **Open in SQL Window** in the **Operation** column.

----End

11.3 How Do I View Slow Query Logs of TaurusDB?

Viewing Log Details

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 **Database** > **TaurusDB**.

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

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

Step 6 On the **Slow Query Logs** page, view the slow query log details.

You can view the slow query log records of a specified execution statement type or a specific time period.

----End

11.4 How Do I Enable Binlog and View Binlog Files of My TaurusDB Instance?

Enabling Binlog

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 **Database** > **TaurusDB**.

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.

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

- 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.

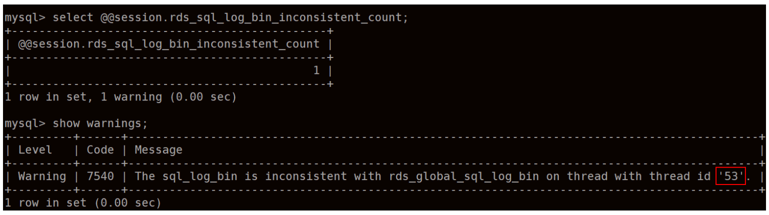
  After this parameter is changed, connect to the database and run the following command to check whether the binlog is enabled for all threads:

  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 11-2 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 11-2**), 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.

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 TaurusDB Performance

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

NOTE

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

11.5 How Do I Change the Binlog Retention Period?

TaurusDB is compatible with the **binlog_expire_logs_seconds** parameter of MySQL Community Edition 8.0. You can change the 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 **Database** > **TaurusDB**.

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

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 **DB Engine Version** field.

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

11.6 Why Are Slow SQL Statements Displayed in Slow Query Log Monitoring, but No Information About Them Is Displayed on the Slow Query Logs Tab Page in the Logs Module?

When **log_slow_admin_statements** is set to **ON**, the database kernel records management SQL statements, such as **Binlog Dump GTID**, **ANALYZE TABLE** and **OPTIMIZE TABLE**, in slow query logs. However, these SQL statements are usually initiated by O&M operations and are not closely related to workloads. So, they are filtered out when uploaded to the **Logs** module, helping users efficiently and accurately check, locate, and analyze slow queries generated on workloads.

12 Version Upgrade

12.1 How Can I Check a TaurusDB Instance Version?

Checking an instance version through the TaurusDB console

Click the target instance to go to the **Basic Information** page. On the **DB Instance Information** area, check the value of **DB Engine Version**.