TaurusDB

TroubleShooting

Issue 01

Date 2024-12-30

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/

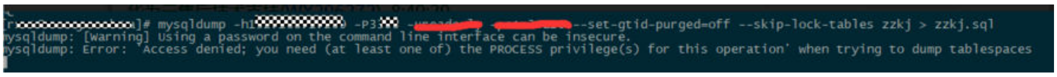
Contents

1 Backup and Restoration Issues

1.1 Insufficient Permissions During Data Export Using mysqldump

Scenario

When you export database data with mysqldump using a specified user account, the error message "Access denied; you need (at least one of) the PROCESS privilege(s)" is displayed.



```
[r[redacted]]# mysqldump -h1[redacted] -P3[redacted] -[redacted] [redacted]--set-gtid-purged=off --skip-lock-tables zzkj > zzkj.sql
ysqldump: [Warning] Using a password on the command line interface can be insecure.
ysqldump: Error: 'Access denied; you need (at least one of) the PROCESS privilege(s) for this operation' when trying to dump tablespaces
```

Possible Causes

The user account does not have the PROCESS permission.

Solution

Grant the PROCESS permission to the user account as the administrator.

GRANT SELECT, PROCESS ON *.* TO 'dump_user'@'%';

FLUSH PRIVILEGES;

1.2 How Do I use mysqlbinlog to Obtain Binlog Files?

Use an ECS-based instance as an example.

1. Install the mysql client on the ECS. For details, see **How Can I Install the mysql Client?**

   NOTE

   TaurusDB is only compatible with MySQL Community Edition 8.0 or later.

2. Download binlog files.

mysqlbinlog -h*xxx* **-u***xxx* **-P***xxx* **-p***xxx* **binlog.***xxxx* **--read-from-remote-server**

Common mysqlbinlog parameters:

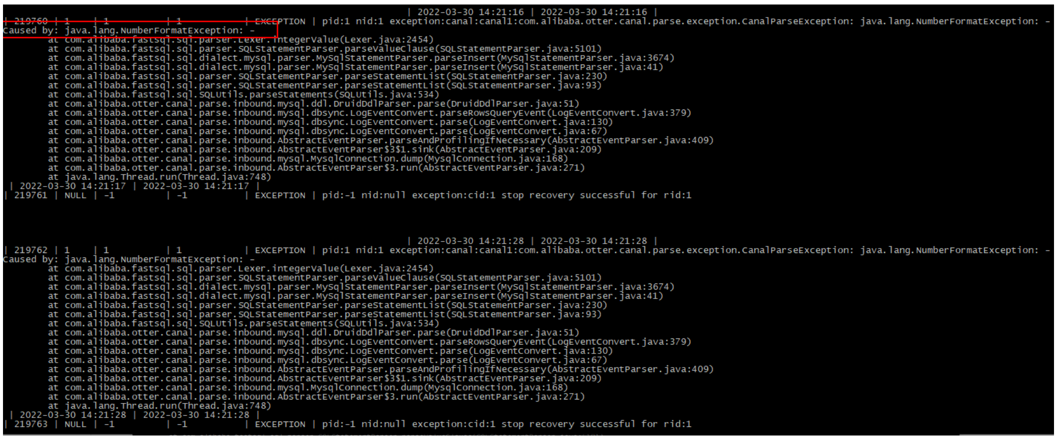
- **-h**: database host.
- **-u**: username.
- **-P**: port number.
- **-p**: password.
- **--start-position**: position where the decoding starts.
- **--start-datetime**: time when the decoding starts.
- **--stop-position**: position where the decoding stops.
- **--stop-datetime**: time where the decoding stops.
- **--skip-gtids**: gtid_log_event is not printed.
- **--short-form**: Only statements are displayed.
- **--result-file**: The SQL file that binlog decoding results are written to.
- **--read-from-remote-server**: Binary logs are read from a remote server (it is available only when mysqlbinlog and the database server are not on the same computer).

1.3 Canal Fails to Parse Binlogs

Scenario

An error occurred when Canal parsed binlogs, interrupting binlog collection. The error message is as follows:

```
com.alibaba.otter.canal.parse.exception.CanalParseException: java.lang.NumberFormatException:- Caused by: java.lang.NumberFormatException:  - at com.alibaba.fastsql.sql.parser.Lexer.integerValue(Lexer.java:2454)
```



```
                                                                                | 2022-03-30 14:21:16 | 2022-03-30 14:21:16 |
| 219760 | 1    | 1            | 1            | EXCEPTION | pid:1 nid:1 exception:canal:canal1:com.alibaba.otter.canal.parse.exception.CanalParseException: java.lang.NumberFormatException: -
Caused by: java.lang.NumberFormatException: -
        at com.alibaba.fastsql.sql.parser.Lexer.integerValue(Lexer.java:2454)
        at com.alibaba.fastsql.sql.parser.SQLStatementParser.parseValueClause(SQLStatementParser.java:5101)
        at com.alibaba.fastsql.sql.dialect.mysql.parser.MySqlStatementParser.parseInsert(MySqlStatementParser.java:3674)
        at com.alibaba.fastsql.sql.dialect.mysql.parser.MySqlStatementParser.parseInsert(MySqlStatementParser.java:41)
        at com.alibaba.fastsql.sql.parser.SQLStatementParser.parseStatementList(SQLStatementParser.java:230)
        at com.alibaba.fastsql.sql.parser.SQLStatementParser.parseStatementList(SQLStatementParser.java:93)
        at com.alibaba.fastsql.sql.SQLUtils.parseStatements(SQLUtils.java:534)
        at com.alibaba.otter.canal.parse.inbound.mysql.ddl.DruidDdlParser.parse(DruidDdlParser.java:51)
        at com.alibaba.otter.canal.parse.inbound.mysql.dbsync.LogEventConvert.parseRowsQueryEvent(LogEventConvert.java:379)
        at com.alibaba.otter.canal.parse.inbound.mysql.dbsync.LogEventConvert.parse(LogEventConvert.java:130)
        at com.alibaba.otter.canal.parse.inbound.mysql.dbsync.LogEventConvert.parse(LogEventConvert.java:67)
        at com.alibaba.otter.canal.parse.inbound.AbstractEventParser.parseAndProfilingIfNecessary(AbstractEventParser.java:409)
        at com.alibaba.otter.canal.parse.inbound.AbstractEventParser$3$1.sink(AbstractEventParser.java:209)
        at com.alibaba.otter.canal.parse.inbound.mysql.MysqlConnection.dump(MysqlConnection.java:168)
        at com.alibaba.otter.canal.parse.inbound.AbstractEventParser$3.run(AbstractEventParser.java:271)
        at java.lang.Thread.run(Thread.java:748)
 | 2022-03-30 14:21:17 | 2022-03-30 14:21:17 |
| 219761 | NULL | -1           | -1           | EXCEPTION | pid:-1 nid:null exception:cid:1 stop recovery successful for rid:1


                                                                                | 2022-03-30 14:21:28 | 2022-03-30 14:21:28 |
| 219762 | 1    | 1            | 1            | EXCEPTION | pid:1 nid:1 exception:canal:canal1:com.alibaba.otter.canal.parse.exception.CanalParseException: java.lang.NumberFormatException: -
Caused by: java.lang.NumberFormatException: -
        at com.alibaba.fastsql.sql.parser.Lexer.integerValue(Lexer.java:2454)
        at com.alibaba.fastsql.sql.parser.SQLStatementParser.parseValueClause(SQLStatementParser.java:5101)
        at com.alibaba.fastsql.sql.dialect.mysql.parser.MySqlStatementParser.parseInsert(MySqlStatementParser.java:3674)
        at com.alibaba.fastsql.sql.dialect.mysql.parser.MySqlStatementParser.parseInsert(MySqlStatementParser.java:41)
        at com.alibaba.fastsql.sql.parser.SQLStatementParser.parseStatementList(SQLStatementParser.java:230)
        at com.alibaba.fastsql.sql.parser.SQLStatementParser.parseStatementList(SQLStatementParser.java:93)
        at com.alibaba.fastsql.sql.SQLUtils.parseStatements(SQLUtils.java:534)
        at com.alibaba.otter.canal.parse.inbound.mysql.ddl.DruidDdlParser.parse(DruidDdlParser.java:51)
        at com.alibaba.otter.canal.parse.inbound.mysql.dbsync.LogEventConvert.parseRowsQueryEvent(LogEventConvert.java:379)
        at com.alibaba.otter.canal.parse.inbound.mysql.dbsync.LogEventConvert.parse(LogEventConvert.java:130)
        at com.alibaba.otter.canal.parse.inbound.mysql.dbsync.LogEventConvert.parse(LogEventConvert.java:67)
        at com.alibaba.otter.canal.parse.inbound.AbstractEventParser.parseAndProfilingIfNecessary(AbstractEventParser.java:409)
        at com.alibaba.otter.canal.parse.inbound.AbstractEventParser$3$1.sink(AbstractEventParser.java:209)
        at com.alibaba.otter.canal.parse.inbound.mysql.MysqlConnection.dump(MysqlConnection.java:168)
        at com.alibaba.otter.canal.parse.inbound.AbstractEventParser$3.run(AbstractEventParser.java:271)
        at java.lang.Thread.run(Thread.java:748)
 | 2022-03-30 14:21:28 | 2022-03-30 14:21:28 |
| 219763 | NULL | -1           | -1           | EXCEPTION | pid:-1 nid:null exception:cid:1 stop recovery successful for rid:1
```

Possible Causes

Check whether the value of **binlog_rows_query_log_events** of your TaurusDB instance is set to **1** or **ON**.

- Canal supports only subscriptions to binlogs in row format.
- When the value of **binlog_rows_query_log_events** is set to **1** or **ON**, Rows_query events are generated in binlogs. These events are not in row

format. In certain scenarios, blank topics may occur in Canal, resulting in a binlog parsing failure.

Solution

Change the value of **binlog_rows_query_log_events** to **OFF** and restart the interrupted Canal task.

1.4 Precautions for Exporting Large Tables Through mysqldump

If the **–q** or **--quick** parameter is added when you use mysqldump to export data, the results of SELECT statements are not buffered in memory but directly exported. If this parameter is disabled, the results of SELECT statements are buffered in memory and then sent to the client.

- If you use mysqldump to back up only a small amount of data which can be stored in the idle memory buffer, disabling **-q** increases the export speed.
- Buffering a large amount of data may consume a large amount of memory, causing a memory swapping. If you use mysqldump to back up a large amount of data which cannot be stored in the memory buffer, enable **-q**. If **-q** is not enabled, a large amount of memory will be consumed and may even cause the database to break down due to out of memory.

Therefore, you are advised to enable the **-q** parameter when using mysqldump to back up data.

Example command:

mysqldump -uroot -p-P*8635* **-h***192.168.0.199* **--set-gtid-purged=OFF --single-transaction --flush-logs -q** *test t1*>t1**.sql**

1.5 Commands for Exporting Data Through mysqldump

Background

mysqldump is the most commonly used tool for importing and exporting MySQL data.

mysqldump Options

Table 1-1 Option description

Option Name	Description
add-drop-table	Adds the DROP TABLE statement before each data table is created.
events, E	Exports events.

Option Name	Description
routines, R	Exports stored procedures and customized functions.
flush-logs	Updates logs before the logs are exported.
no-create-db, n	Exports only data without adding of the CREATE DATABASE statement.
add-drop-database	Adds the DROP DATABASE statement before each database is created.
no-create-info, t	Exports only data without adding of the CREATE TABLE statement.
no-data, d	Exports only table structure data.
set-gtid-purged=OFF	Does not export GTID statements.
hex-blob	Exports binary string fields in hexadecimal format.

Scenario

Examples are as follows:

1. Export all data of databases **db1** and **db2**.

   ```
   mysqldump -uroot -p -P8635 -h192.168.0.199 --hex-blob --set-gtid-purged=OFF --single-transaction --order-by-primary --flush-logs -q --databases db1 db2 >db12.sql
   ```

2. Export the **t1** and **t2** tables of database **db1**.

   ```
   mysqldump -uroot -p -P8635 -h192.168.0.199 --hex-blob --set-gtid-purged=OFF --single-transaction --order-by-primary --flush-logs -q --databases db1 --tables t1 t2 >t1_t2.sql
   ```

3. Export data whose id equals **1** from table **t1** in database **db1**.

   ```
   mysqldump -uroot -p -P8635 -h192.168.0.199 --hex-blob --set-gtid-purged=OFF --single-transaction --order-by-primary --flush-logs -q --databases db1 --tables t1 --where='id=1'>t1_id.sql
   ```

4. Export all table structures in database **db1** without exporting data.

   ```
   mysqldump -uroot -p -P8635 -h192.168.0.199 --no-data --set-gtid-purged=OFF --single-transaction --order-by-primary -n --flush-logs -q --databases db1 >db1_table.sql
   ```

5. Export all data excluding the tables and data in database **db1**.

   ```
   mysqldump -uroot -p -h192.168.0.199 -P8635 --set-gtid-purged=OFF -F -n -t -d -E -R db1> others.sql
   ```

1.6 System Inaccessible After Field Addition to a Database Table

Scenarios

After a field was added to a database table, the system becomes inaccessible.

Solution

The database performance is affected due to the addition of table fields. A possible reason is that indexes are not added to the new table fields. As a result, a large amount of data consumes a large number of CPU resources. You are advised to:

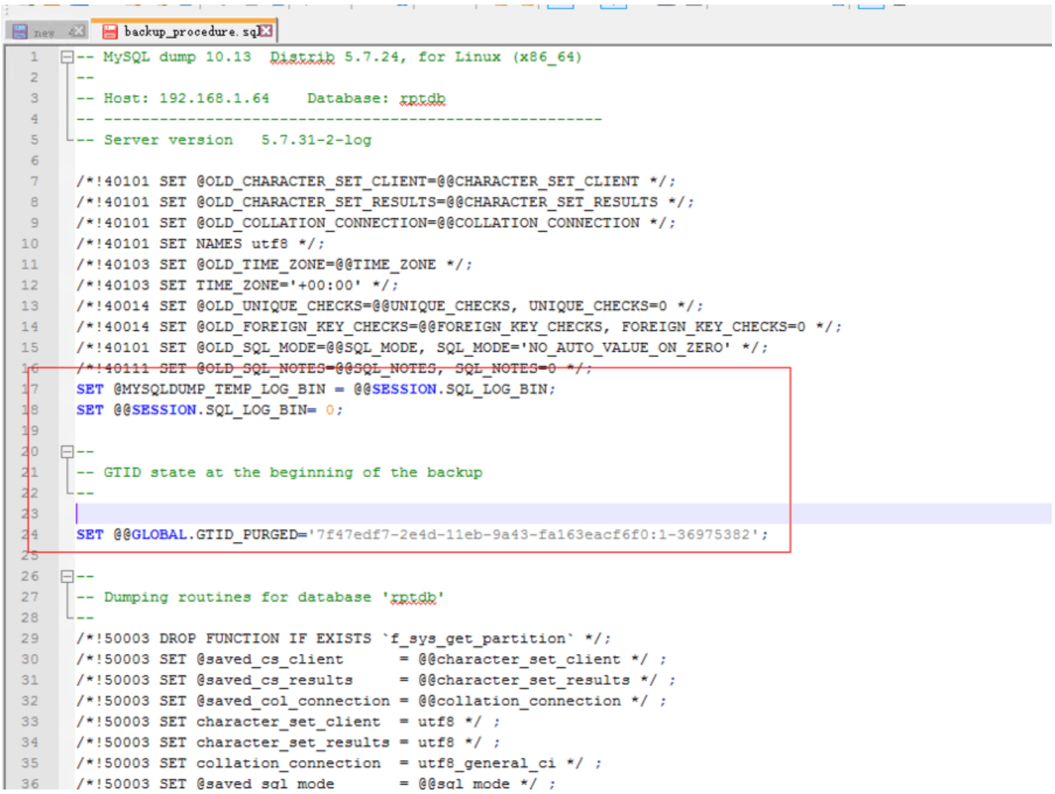
- Add indexes and primary keys.
- Optimize slow SQL statements.

1.7 SQL Statements Such as SET @@SESSION.SQL_LOG_BIN Displayed After You Run mysqldump

Scenario

When you run mysqldump on a newly purchased Huawei Cloud database, the following code is displayed.

Figure 1-1 Code



```
-- MySQL dump 10.13  Distrib 5.7.24, for Linux (x86_64)
--
-- Host: 192.168.1.64    Database: rptdb
-- ------------------------------------------------------
-- Server version	5.7.31-2-log

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;
SET @MYSQLDUMP_TEMP_LOG_BIN = @@SESSION.SQL_LOG_BIN;
SET @@SESSION.SQL_LOG_BIN= 0;

--
-- GTID state at the beginning of the backup
--

SET @@GLOBAL.GTID_PURGED='7f47edf7-2e4d-11eb-9a43-fa163eacf6f0:1-36975382';

--
-- Dumping routines for database 'rptdb'
--
/*!50003 DROP FUNCTION IF EXISTS `f_sys_get_partition` */;
/*!50003 SET @saved_cs_client      = @@character_set_client */ ;
/*!50003 SET @saved_cs_results     = @@character_set_results */ ;
/*!50003 SET @saved_col_connection = @@collation_connection */ ;
/*!50003 SET character_set_client  = utf8 */ ;
/*!50003 SET character_set_results = utf8 */ ;
/*!50003 SET collation_connection  = utf8_general_ci */ ;
/*!50003 SET @saved_sql_mode       = @@sql_mode */ ;
```

Fault Analysis

The parameter **gtid-mode** is set to **ON**.

If GTID is enabled for a database, you can use mysqldump to back up or dump all Global Transaction Identifiers (GTIDs) in the database or even to back up the whole MySQL database.

Solution

When TaurusDB databases are exported for backup and restoration, check whether GTID is enabled.

If GTID is enabled, add **–set-gtid-purged=OFF** to the **mysqldump** command during data dump.

1.8 Insufficient Privileges Reported for Canal

Scenario

When you start Canal while obtaining binlogs from TaurusDB using a specified user account, the following error message is often displayed: 'show master status' has an error! Access denied: you need (at least one of) the SUPER, REPLICATION CLIENT privilege(s) for this operation.

The complete error information is as follows:

```
2021-01-10 23:58:32.964 [destination = evoicedc , address = /dbus-mysql:3306 , EventParser] ERROR
com.alibaba.ot ter.canal.common.alarm.LogAlarmHandler -
destination:evoicedc[com.alibaba.otter.canal.parse.exception.CanalParseEx ception: command : 'show master
status' has an error!
Caused by: java.io.IOException: ErrorPacket [errorNumber=1227, fieldCount=-1, message=Access denied;
you need (at least one of) the SUPER, REPLICATION CLIENT privilege(s) for this operation, sqlState=42000,
sqlStateMarker=#] with command: show master status at
com.alibaba.otter.canal.parse.driver.mysql.MysqlQueryExecutor.query(MysqlQueryExecutor.java:61)
```

Possible Causes

The user account does not have the REPLICATION SLAVE or REPLICATION CLIENT permissions.

Solution

Grant the REPLICATION SLAVE and REPLICATION CLIENT permissions to the user account as the administrator.

GRANT SELECT, REPLICATION SLAVE, REPLICATION CLIENT ON *.* TO 'canal'@'%';

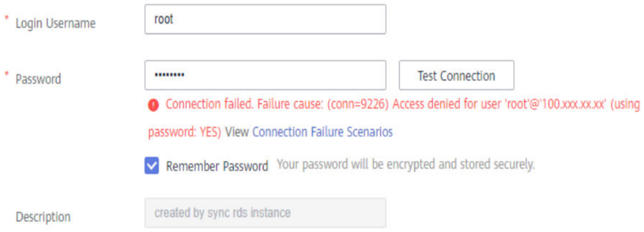
FLUSH PRIVILEGES;

2 Connection Issues

2.1 Login Failed After ssl_type of root Is Changed to ANY

Scenario

When user **root** was used to log in to a DB instance through DAS on the console, the error message "Access denied" was displayed.



* Login Username: root

* Password: ••••••••• Test Connection

Connection failed. Failure cause: (conn=9226) Access denied for user 'root'@'100.xxx.xx.xx' (using password: YES) View Connection Failure Scenarios

☑ Remember Password Your password will be encrypted and stored securely.

Description: created by sync rds instance

Possible Causes

1. View the **root** account information in the **mysql.user** table to check whether the client IP address is within the allowed range and whether SSL is enabled.

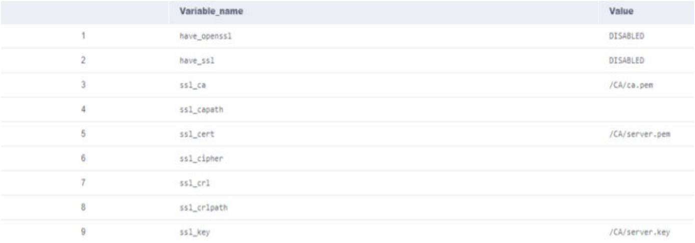
   ```
   SELECT * FROM mysql.user WHERE User='root';
   ```

   If **ssl_type** of the **root** account is set to **ANY**, the **root** account needs to use SSL.

2. Check whether SSL is enabled.

   show variables like '%ssl%';

   SSL was not enabled for the instance.



	Variable_name	Value
1	have_openssl	DISABLED
2	have_ssl	DISABLED
3	ssl_ca	/CA/ca.pem
4	ssl_capath	
5	ssl_cert	/CA/server.pem
6	ssl_cipher	
7	ssl_crl	
8	ssl_crlpath	
9	ssl_key	/CA/server.key

The cause is that **ssl_type** of the **root** account was changed to **ANY**. As a result, the login failed.

Solution

Run the following command to change the value of **ssl_type** to be empty for the **root** account:

update mysql.user set ssl_type='' where user = 'root';

To change the **ssl_type** of all other user accounts to be empty, run the following command:

update mysql.user set ssl_type='' where user not like 'mysql%';

2.2 Failed to Connect to a DB Instance Using SSL

Scenario

Your DB instance cannot be connected using SSL.

Possible Causes

There may be a network connection issue. If your DB instance can be connected without SSL, the mysql client or the corresponding database driver version may be incompatible with your DB instance.

Solution

TaurusDB is compatible with MySQL Community Edition 8.0 or later. Install the mysql client or database driver 8.0 or later.

Secure Socket Layer (SSL) uses data encryption, identity verification, and message integrity check to ensure connection security.

SSL provides the following functions:

1. Data encryption: A symmetric key algorithm encrypts data in transit.
2. Identity verification: Digital signatures authenticate clients and servers based on certificates.
3. Message integrity check: A MAC algorithm checks the integrity of messages in transit.

Notice

- If SSL is enabled on the server, the client can connect to the server without using SSL and data is not encrypted.
- If SSL is not used, data is transmitted in plaintext and there are potential security risks.
- SSL is enabled by default for TaurusDB instances. For details about how to disable SSL, see **Configuring SSL for a DB Instance**.
- For details about how to use SSL for the mysql client connection, see **Connecting to a DB Instance Through the mysql Client**.

- For details about how to use SSL for a JDBC connection, see **Connecting to a DB Instance Through JDBC**.

2.3 Description of Each IP Address

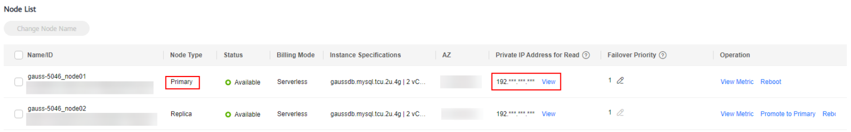
If you create a TaurusDB instance with one primary node and one read replica, there are up to five different IP addresses.

NOTE

If you connect your application to a node of the DB instance through a private IP address for read and the node becomes faulty, the IP address cannot be accessed before the fault is rectified.

1. Private IP address for read of the primary node (not recommended)

   After a DB instance is created, the system assigns a private IP address for read to the primary node. If your applications are deployed on a server that is in the same VPC as the DB instance, you can use the IP address to connect to the instance for read and write operations. But if a failover occurs and the primary node becomes a read replica, the IP address can only be used for read operations.



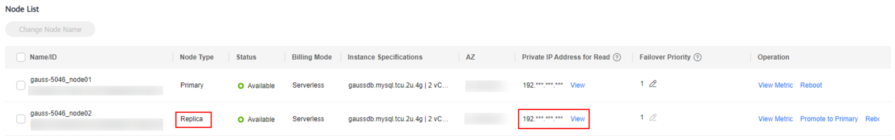
Node List

Change Node Name

Name/ID	Node Type	Status	Billing Mode	Instance Specifications	AZ	Private IP Address for Read	Failover Priority	Operation
□ gauss-5046_node01 [redacted]	Primary	Available	Serverless	gaussdb.mysql.tcu.2u.4g | 2 vC...	[redacted]	192.***.***.*** View	1	View Metric Reboot
□ gauss-5046_node02 [redacted]	Replica	Available	Serverless	gaussdb.mysql.tcu.2u.4g | 2 vC...	[redacted]	192.***.***.*** View	1	View Metric Promote to Primary Reb

2. Private IP address for read of a read replica (not recommended).

   After a DB instance is created, the system assigns a private IP address to the read replica. If your applications are deployed on a server that is in the same VPC as the DB instance, you can use this IP address to connect to the instance for read operations. But if a failover occurs and the read replica is promoted to primary, the IP address can be used for read and write operations.



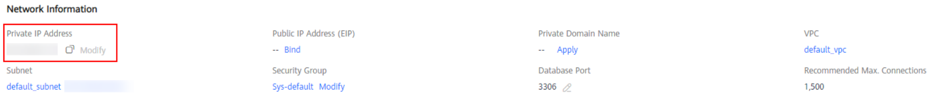
Node List

Change Node Name

Name/ID	Node Type	Status	Billing Mode	Instance Specifications	AZ	Private IP Address for Read	Failover Priority	Operation
□ gauss-5046_node01 [redacted]	Primary	Available	Serverless	gaussdb.mysql.tcu.2u.4g | 2 vC...	[redacted]	192.***.***.*** View	1	View Metric Reboot
□ gauss-5046_node02 [redacted]	Replica	Available	Serverless	gaussdb.mysql.tcu.2u.4g | 2 vC...	[redacted]	192.***.***.*** View	1	View Metric Promote to Primary Reb

3. Private IP address

   After a DB instance is created, the system assigns a private IP address to the primary node. If your applications are deployed on a server that is in the same VPC as the DB instance, you can use the IP address to connect to the instance for read and write operations. But this IP address is always bound to the primary node. If a failover occurs, the IP address is reassigned to the new primary node, and can then still be used for read and write operations.



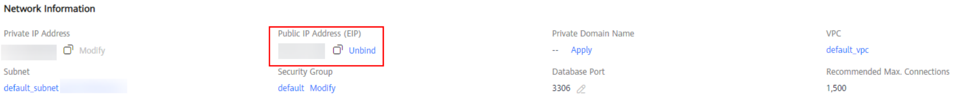
Network Information

Private IP Address [redacted] Modify	Public IP Address (EIP) -- Bind	Private Domain Name -- Apply	VPC default_vpc
Subnet default_subnet [redacted]	Security Group Sys-default Modify	Database Port 3306	Recommended Max. Connections 1,500

4. Public IP address (EIP)

   After you buy a DB instance, you can bind an EIP to the instance to enable public accessibility, but can also unbind it later if needed. Just like a private IP address, an EIP is always bound to the primary node for read and write operations.

Figure 2-1 Viewing the EIP and database port

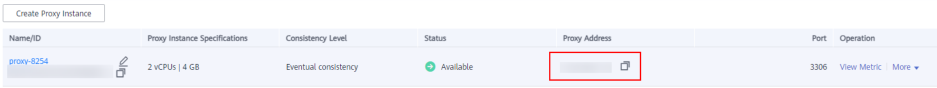


Network Information

Private IP Address	Public IP Address (EIP)	Private Domain Name	VPC
[redacted] Modify	[redacted] Unbind	-- Apply	default_vpc
Subnet	Security Group	Database Port	Recommended Max. Connections
default_subnet [redacted]	default Modify	3306	1,500

5. Proxy address

   After you buy a DB instance, you can enable database proxy and create a proxy instance. Then, the system assigns an IP address to the proxy instance. The proxy instance sends write requests to the primary node and read requests to the read replica based on the IP address, offloading the pressure on the primary node. Currently, read/write splitting IP addresses can only be used for intranet access.



Create Proxy Instance

Name/ID	Proxy Instance Specifications	Consistency Level	Status	Proxy Address	Port	Operation
proxy-8254 [redacted]	2 vCPUs | 4 GB	Eventual consistency	Available	[redacted]	3306	View Metric | More

NOTE

Failover

By default, a TaurusDB instance contains at least two nodes, one primary node (read/write node) and one read replica (read-only node). You can create multiple read replicas if needed.

If the primary node becomes faulty, the system promotes a read replica to primary and demotes the primary node to read replica status.

2.4 SSL Connection Failed Due to Inconsistent TLS Versions

Scenario

A client failed to connect to a cloud-based TaurusDB instance using SSL, but could connect to a self-managed database using SSL.

Possible Causes

Troubleshooting:

1. **View error logs of the TaurusDB instance.**

```
2021-07-09T10:30:58.476586+08:00 212539 [Warning] SSL errno: 337678594, SSL errmsg:
error:14209102:SSL routines:tls_early_post_process_client_hello:unsupported
protocol2021-07-09T10:30:58.476647+08:00 212539 [Note] Bad
handshake2021-07-09T10:32:43.535738+08:00 212631 [Warning] SSL errno: 337678594, SSL errmsg:
error:14209102:SSL routines:tls_early_post_process_client_hello:unsupported
protocol2021-07-09T10:32:43.535787+08:00 212631 [Note] Bad
handshake2021-07-09T10:50:03.401100+08:00 213499 [Warning] SSL errno: 337678594, SSL errmsg:
error:14209102:SSL routines:tls_early_post_process_client_hello:unsupported
protocol2021-07-09T10:50:03.401161+08:00 213499 [Note] Bad
handshake2021-07-09T10:53:44.458404+08:00 213688 [Warning] SSL errno: 337678594, SSL errmsg:
error:14209102:SSL routines:tls_early_post_process_client_hello:unsupported
protocol2021-07-09T10:53:44.458475+08:00 213688 [Note] Bad handshake
```

2. According to **unsupported protocol** in the error information, the problem may be related to the TLS version. Run the following command to check the TLS versions of the TaurusDB instance and self-managed database:

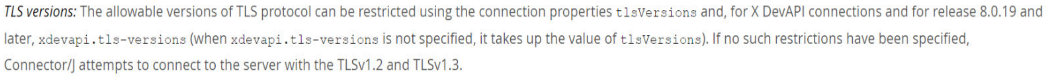
   show variables like '%tls_version%';

   It was found that the TaurusDB instance used TLS v1.2 and the self-managed database used TLS v1.1. The TLS version of the client was the same as that of the self-managed database. As a result, the self-managed database was successfully connected, but the TaurusDB instance failed to be connected.

Solution

Upgrade the TLS version of the client to TLS v1.2.

If the official JDBC driver **mysql-connector/J** is used, see **Connecting Securely Using SSL** for the configuration method.



TLS versions: The allowable versions of TLS protocol can be restricted using the connection properties `tlsVersions` and, for X DevAPI connections and for release 8.0.19 and later, `xdevapi.tls-versions` (when `xdevapi.tls-versions` is not specified, it takes up the value of `tlsVersions`). If no such restrictions have been specified, Connector/J attempts to connect to the server with the TLSv1.2 and TLSv1.3.

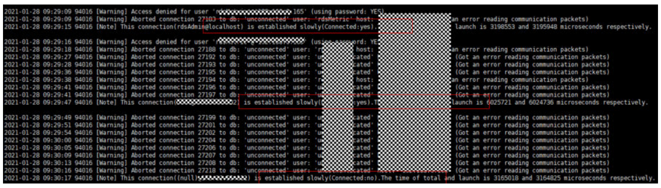
2.5 Error Message "connection established slowly"

Scenario

During peak hours, the connection between a client and a TaurusDB instance often times out. As a result, it takes more than 10 seconds to log in to the instance.

Possible Causes

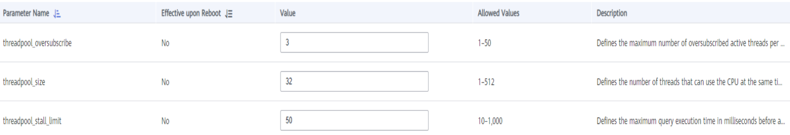
1. **View error logs of the TaurusDB instance** to check whether the error message "connection xxx is established slowly" is displayed. Example:

```
2021-01-28 09:29:09 94016 [Warning] Access denied for user '[redacted]165' (using password: YES)
2021-01-28 09:29:09 94016 [Warning] Aborted connection 27103 to db: 'unconnected' user: 'rdsMetric' host: [redacted]an error reading communication packets)
2021-01-28 09:29:15 94016 [Note] This connection(rdsAdmin@localhost) is established slowly(Connected:yes).[redacted] launch is 3198553 and 3195948 microseconds respectively.

2021-01-28 09:29:16 94016 [Warning] Access denied for user '[redacted]' (using password: YE[redacted]
2021-01-28 09:29:18 94016 [Warning] Aborted connection 27188 to db: 'unconnected' user: 'r[redacted] host: [redacted]an error reading communication packets)
2021-01-28 09:29:27 94016 [Warning] Aborted connection 27192 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:29:28 94016 [Warning] Aborted connection 27193 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:29:36 94016 [Warning] Aborted connection 27195 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:29:38 94016 [Warning] Aborted connection 27194 to db: 'unconnected' user: 'r[redacted] host: [redacted]an error reading communication packets)
2021-01-28 09:29:41 94016 [Warning] Aborted connection 27196 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:29:41 94016 [Warning] Aborted connection 27197 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:29:47 94016 [Note] This connection([redacted]2) is established slowly([redacted]:yes).T[redacted]launch is 6025721 and 6024736 microseconds respectively.

2021-01-28 09:29:49 94016 [Warning] Aborted connection 27199 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:29:51 94016 [Warning] Aborted connection 27201 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:29:54 94016 [Warning] Aborted connection 27202 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:30:00 94016 [Warning] Aborted connection 27204 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:30:05 94016 [Warning] Aborted connection 27206 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:30:09 94016 [Warning] Aborted connection 27207 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:30:13 94016 [Warning] Aborted connection 27208 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:30:16 94016 [Warning] Aborted connection 27218 to db: 'unconnected' user: 'u[redacted]cated' [redacted] (Got an error reading communication packets)
2021-01-28 09:30:17 94016 [Note] This connection((null)[redacted]) is established slowly(Connected:no).The time of total and launch is 3165018 and 3164825 microseconds respectively.
```

   If so, some connections have timed out and have not been processed by the instance yet. When the connection between the client and the instance exceeds the specified timeout duration, an error is reported.

2. Check the thread pool configuration (enabled by default) on the console.

Parameter Name	Effective upon Reboot	Value	Allowed Values	Description
threadpool_oversubscribe	No	3	1-50	Defines the maximum number of oversubscribed active threads per ...
threadpool_size	No	32	1-512	Defines the number of threads that can use the CPU at the same ti...
threadpool_stall_limit	No	50	10-1,000	Defines the maximum query execution time in milliseconds before a...

   In the preceding figure, **threadpool_size** is set to **1**, **threadpool_stall_limit** is set to **500 ms**, and **threadpool_oversubscribe** is set to **3**. The wait time for the thread pool to process connections is mainly related to the three parameters.

- When all threads in the thread pool are busy, the scheduling thread in the thread pool creates a new thread every 500 ms (**threadpool_stall_limit**). Each thread group can process a new connection every 500 ms on average. If the queue is too long, the client may time out.
- If all threads in the thread pool are busy, it means that the number of the threads has reached the upper limit. So when there are a large number of connections to be established, the number of total threads is calculated as follows: **threadpool_size** x (**threadpool_oversubscribe** + 1)

Solution

If there are a large number of new connections, increase the value of **threadpool_oversubscribe** to increase the total number of threads.

This reduces the overhead of repeated thread creation and destruction, and limits the number of running threads of the TaurusDB instance, thus protecting the system from crashing.

In normal cases, the thread pool is used when there are a large number of short connections. If persistent connections are used and there are a few connections (for example, the client uses a connection pool), the thread pool is not so helpful. In this case, adjust the values of **threadpool_size** and **threadpool_oversubscribe** to increase the total number of threads, or close the thread pool.

2.6 "Access denied" Displayed During Database Connection

Scenario

A client failed to connect to a database, and the error message "Error 1045: Access denied for user xxx" was displayed.

Handling Methods

1. An incorrect host is connected.

   Cause: An incorrect database host is connected, and the user or client IP address does not have the access permission.

   Solution: Ensure that the host name of the database to be connected is correctly specified.

2. The user does not exist.

   Cause: The user account used for connecting to the database does not exist.

   Solution:

   - Log in to the database as an administrator and run the following command to check whether the target user exists:
     ```
     SELECT User FROM mysql.user WHERE User='xxx';
     ```
   - If the user does not exist, create the user.
     ```
     CREATE USER 'xxx'@'xxxxxx' IDENTIFIED BY 'xxxx';
     ```

3. The client IP address does not have the access permission.

Cause: The user used by the client exists, but the client IP address is not allowed to access the database.

Solution:

- Log in to the database as an administrator and run the following command to check which client IP addresses are allowed to connect to the database for the target user:

  ```
  SELECT Host, User FROM mysql.user WHERE User='xxx';
  ```

- If the client IP address is not within the allowed network segment, assign the access permission to the client IP address. For example, run the following command to grant the **test** user the permission to access the **192.168.0** network segment:

  ```
  GRANT ALL PRIVILEGES ON *.* TO'root'@'192.168.0.%' IDENTIFIED BY 'password' WITH GRANT
  OPTION;
  FLUSH PRIVILEGES;
  ```

4. The password is incorrect.

   Cause: The password of the user is incorrect.

   Solution:

   - Check whether the password is correct. As the password is used for identity authentication, it cannot be retrieved in plain text from TaurusDB. However, you can compare the hash string with the **PASSWORD** function value of the password to check whether the password is correct. The following is an example of SQL statements:

     ```
     mysql> SELECT Host, User, authentication_string, PASSWORD('12345') FROM mysql.user
     WHERE User='test';
     +-----------+------+-------------------------------------------+-------------------------------------------+
     | Host      | User | authentication_string                     | PASSWORD('12345')                         |
     +-----------+------+-------------------------------------------+-------------------------------------------+
     | %         | test | *6A23DC5E7446019DC9C1778554ED87BE6BA61041 |
     *00A51F3F48415C7D4E8908980D443C29C69B60C9 |
     +-----------+------+-------------------------------------------+-------------------------------------------+
     2 rows in set, 1 warning (0.00 sec)
     ```

     The preceding example shows that the hash value of **PASSWORD('12345')** does not match the **authentication_string** field, indicating that the target password **12345** is incorrect.

   - To reset the user password, run the following SQL statement:

     ```
     set password for 'test'@'%' = 'new_password';
     ```

5. The password contains special characters and is escaped by Bash.

   Cause: In the default Bash environment of Linux, when the CLI is used to connect to a database, special characters in the password will be escaped by the environment. As a result, the password becomes invalid.

   For example, in the Bash environment, the password of user **test** is **test$123**. When you run the **mysql -hxxx -u test -ptest$123** command to connect to a database, the error message "ERROR 1045 (28000): Access denied" will be displayed.

   Solution: Enclose the password in single quotation marks to prevent Bash from interpreting special characters.

   ```
   mysql -hxxx -u test -p'test$123'
   ```

6. **REQUIRE SSL** is configured for the user, but the client uses a non-SSL connection.

   Troubleshooting:

- Run the **show create user 'xxx'** command to check whether the user must use the SSL connection. If the **REQUIRE SSL** attribute is displayed, the user must use the SSL connection.
- Check whether statements similar to the following have been used to grant permissions to the user:
  ```
  GRANT ALL PRIVILEGES ON . TO 'ssluser'@'localhost' IDENTIFIED BY 'password' REQUIRE SSL;
  ```
- Check the **ssl_type** value of the target user. If the value is not empty, the user must use SSL.
  ```
  SELECT User, Host, ssl_type FROM mysql.user WHERE User='xxx';
  ```

Solution:

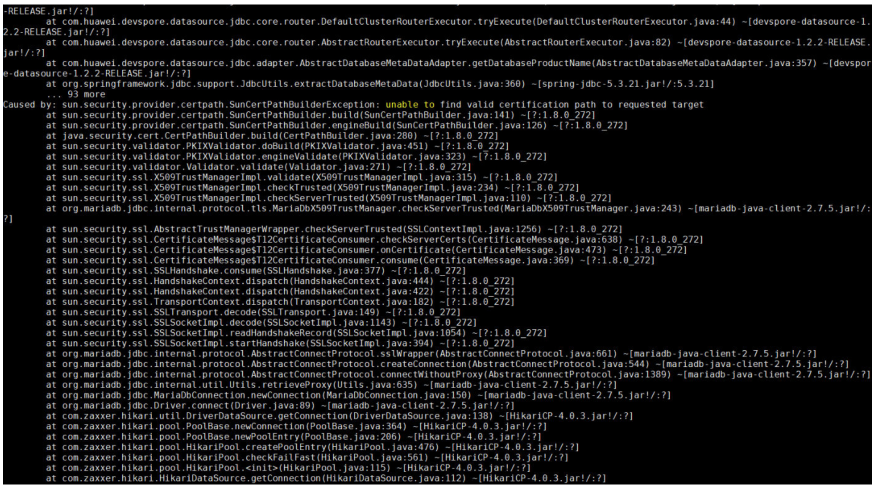
- Connect the client to the database in SSL mode. For details, see **Using a MySQL Client to Connect to a DB Instance Over a Private Network**.
- Run the **ALTER USER 'username'@'host' REQUIRE NONE;** command to remove the SSL permission from the user.

2.7 Failed to Connect to a Database Using mariadb-connector in SSL Mode

Scenario

A database could not be connected using JDBC, and the following error message was displayed:

```
unable to find certification path to requested target
```



```
-RELEASE.jar!/:?]
        at com.huawei.devspore.datasource.jdbc.core.router.DefaultClusterRouterExecutor.tryExecute(DefaultClusterRouterExecutor.java:44) ~[devspore-datasource-1.
2.2-RELEASE.jar!/:?]
        at com.huawei.devspore.datasource.jdbc.core.router.AbstractRouterExecutor.tryExecute(AbstractRouterExecutor.java:82) ~[devspore-datasource-1.2.2-RELEASE.
jar!/:?]
        at com.huawei.devspore.datasource.jdbc.adapter.AbstractDatabaseMetaDataAdapter.getDatabaseProductName(AbstractDatabaseMetaDataAdapter.java:357) ~[devspor
e-datasource-1.2.2-RELEASE.jar!/:?]
        at org.springframework.jdbc.support.JdbcUtils.extractDatabaseMetaData(JdbcUtils.java:360) ~[spring-jdbc-5.3.21.jar!/:5.3.21]
        ... 93 more
Caused by: sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested target
        at sun.security.provider.certpath.SunCertPathBuilder.build(SunCertPathBuilder.java:141) ~[?:1.8.0_272]
        at sun.security.provider.certpath.SunCertPathBuilder.engineBuild(SunCertPathBuilder.java:126) ~[?:1.8.0_272]
        at java.security.cert.CertPathBuilder.build(CertPathBuilder.java:280) ~[?:1.8.0_272]
        at sun.security.validator.PKIXValidator.doBuild(PKIXValidator.java:451) ~[?:1.8.0_272]
        at sun.security.validator.PKIXValidator.engineValidate(PKIXValidator.java:323) ~[?:1.8.0_272]
        at sun.security.validator.Validator.validate(Validator.java:271) ~[?:1.8.0_272]
        at sun.security.ssl.X509TrustManagerImpl.validate(X509TrustManagerImpl.java:315) ~[?:1.8.0_272]
        at sun.security.ssl.X509TrustManagerImpl.checkTrusted(X509TrustManagerImpl.java:234) ~[?:1.8.0_272]
        at sun.security.ssl.X509TrustManagerImpl.checkServerTrusted(X509TrustManagerImpl.java:110) ~[?:1.8.0_272]
        at org.mariadb.jdbc.internal.protocol.tls.MariaDbX509TrustManager.checkServerTrusted(MariaDbX509TrustManager.java:243) ~[mariadb-java-client-2.7.5.jar!/:
?]

        at sun.security.ssl.AbstractTrustManagerWrapper.checkServerTrusted(SSLContextImpl.java:1256) ~[?:1.8.0_272]
        at sun.security.ssl.CertificateMessage$T12CertificateConsumer.checkServerCerts(CertificateMessage.java:638) ~[?:1.8.0_272]
        at sun.security.ssl.CertificateMessage$T12CertificateConsumer.onCertificate(CertificateMessage.java:473) ~[?:1.8.0_272]
        at sun.security.ssl.CertificateMessage$T12CertificateConsumer.consume(CertificateMessage.java:369) ~[?:1.8.0_272]
        at sun.security.ssl.SSLHandshake.consume(SSLHandshake.java:377) ~[?:1.8.0_272]
        at sun.security.ssl.HandshakeContext.dispatch(HandshakeContext.java:444) ~[?:1.8.0_272]
        at sun.security.ssl.HandshakeContext.dispatch(HandshakeContext.java:422) ~[?:1.8.0_272]
        at sun.security.ssl.TransportContext.dispatch(TransportContext.java:182) ~[?:1.8.0_272]
        at sun.security.ssl.SSLTransport.decode(SSLTransport.java:149) ~[?:1.8.0_272]
        at sun.security.ssl.SSLSocketImpl.decode(SSLSocketImpl.java:1143) ~[?:1.8.0_272]
        at sun.security.ssl.SSLSocketImpl.readHandshakeRecord(SSLSocketImpl.java:1054) ~[?:1.8.0_272]
        at sun.security.ssl.SSLSocketImpl.startHandshake(SSLSocketImpl.java:394) ~[?:1.8.0_272]
        at org.mariadb.jdbc.internal.protocol.AbstractConnectProtocol.sslWrapper(AbstractConnectProtocol.java:661) ~[mariadb-java-client-2.7.5.jar!/:?]
        at org.mariadb.jdbc.internal.protocol.AbstractConnectProtocol.createConnection(AbstractConnectProtocol.java:544) ~[mariadb-java-client-2.7.5.jar!/:?]
        at org.mariadb.jdbc.internal.protocol.AbstractConnectProtocol.connectWithoutProxy(AbstractConnectProtocol.java:1389) ~[mariadb-java-client-2.7.5.jar!/:?]
        at org.mariadb.jdbc.internal.util.Utils.retrieveProxy(Utils.java:635) ~[mariadb-java-client-2.7.5.jar!/:?]
        at org.mariadb.jdbc.MariaDbConnection.newConnection(MariaDbConnection.java:150) ~[mariadb-java-client-2.7.5.jar!/:?]
        at org.mariadb.jdbc.Driver.connect(Driver.java:89) ~[mariadb-java-client-2.7.5.jar!/:?]
        at com.zaxxer.hikari.util.DriverDataSource.getConnection(DriverDataSource.java:138) ~[HikariCP-4.0.3.jar!/:?]
        at com.zaxxer.hikari.pool.PoolBase.newConnection(PoolBase.java:364) ~[HikariCP-4.0.3.jar!/:?]
        at com.zaxxer.hikari.pool.PoolBase.newPoolEntry(PoolBase.java:206) ~[HikariCP-4.0.3.jar!/:?]
        at com.zaxxer.hikari.pool.HikariPool.createPoolEntry(HikariPool.java:476) ~[HikariCP-4.0.3.jar!/:?]
        at com.zaxxer.hikari.pool.HikariPool.checkFailFast(HikariPool.java:561) ~[HikariCP-4.0.3.jar!/:?]
        at com.zaxxer.hikari.pool.HikariPool.<init>(HikariPool.java:115) ~[HikariCP-4.0.3.jar!/:?]
        at com.zaxxer.hikari.HikariDataSource.getConnection(HikariDataSource.java:112) ~[HikariCP-4.0.3.jar!/:?]
```

Possible Causes

As shown in the figure above, the JAR package of MariaDB is used to connect to the database, which is slightly different from the official driver package of MySQL.

Solution

The connection string for MariaDB is as follows:

```
String url = "jdbc:mysql://xxx.xxx.xxx.xxx:xxxx/mysql?useSsl=true&serverSslCert=D:\
\ca.pem&disableSslHostnameVerification=true";
```

Note: TaurusDB instances do not support hostname verification. So, you need to set **disableSslHostnameVerification** to **true**. The setting method varies depending on the MariaDB JAR package version. For details, see the **notes on usage** of the corresponding version.

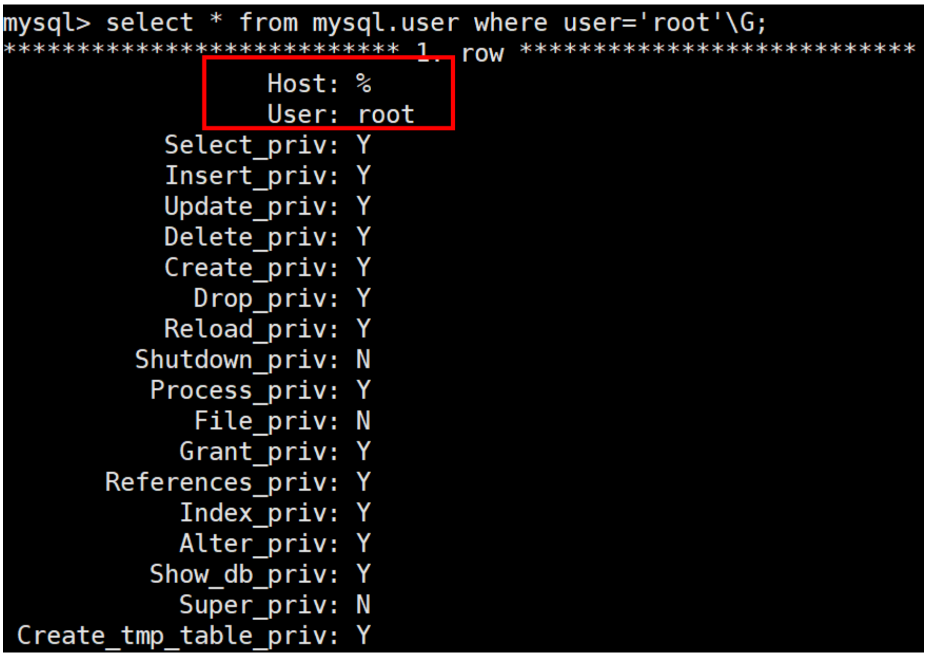
2.8 Failed to Connect to a Database as User root

Scenario

A database failed to be connected using the **root** account.

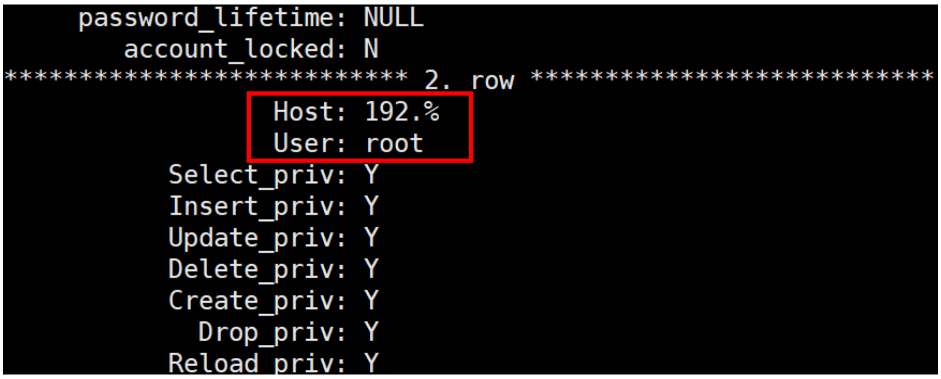
Possible Causes

1. View the kernel error.log to check whether any connection denial records exist.



```
mysql> select * from mysql.user where user='root'\G;
*************************** 1. row ***************************
                  Host: %
                  User: root
           Select_priv: Y
           Insert_priv: Y
           Update_priv: Y
           Delete_priv: Y
           Create_priv: Y
             Drop_priv: Y
           Reload_priv: Y
         Shutdown_priv: N
          Process_priv: Y
             File_priv: N
            Grant_priv: Y
       References_priv: Y
            Index_priv: Y
            Alter_priv: Y
          Show_db_priv: Y
            Super_priv: N
 Create_tmp_table_priv: Y
```

2. Check the **root** permissions. There are two **root** accounts. One of them is allowed to access only hosts whose IP addresses start with 192.



```
     password_lifetime: NULL
        account_locked: N
*************************** 2. row ***************************
                  Host: 192.%
                  User: root
           Select_priv: Y
           Insert_priv: Y
           Update_priv: Y
           Delete_priv: Y
           Create_priv: Y
             Drop_priv: Y
           Reload_priv: Y
```

Solution

Contact Huawei Cloud customer service to delete the extra **root** account.

2.9 Client Automatically Disconnected from a DB Instance

Scenarios

The mysql client was automatically disconnected from the DB instance. The following error information is displayed: ERROR 2013: Lost connection to MySQL server during query.

Solution

ERROR 2013 is usually caused by incorrect configuration.

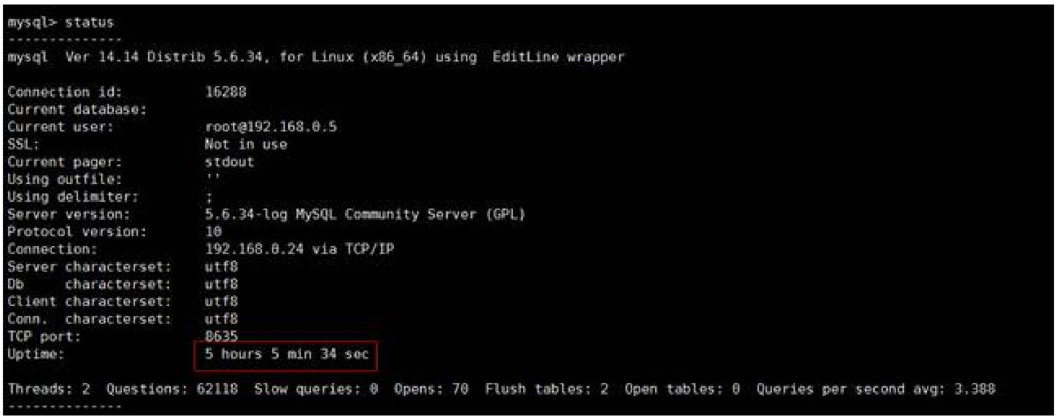
- **wait_timeout**: indicates the number of seconds the server waits for activity on a non-interactive connection before closing it.
- **interactive_timeout**: indicates the number of seconds the server waits for activity on an interactive connection before closing it.

Step 1 Check whether the DB instance is available.

If the DB instance is available, check for other possible causes.

Step 2 View error logs.

Step 3 Use the mysql command-line client to connect to the DB instance. Run **status** to check whether the DB instance has been rebooted frequently.



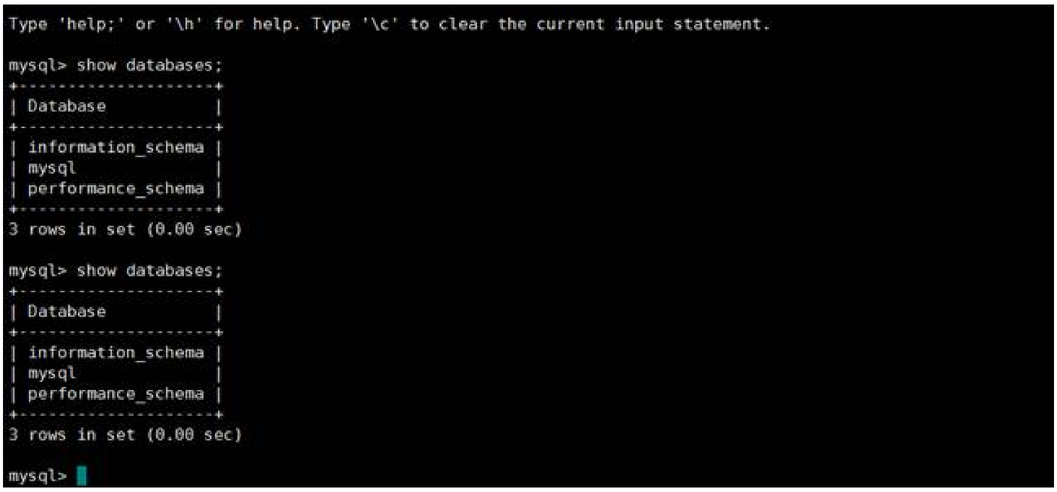
```
mysql> status
--------------
mysql  Ver 14.14 Distrib 5.6.34, for Linux (x86_64) using  EditLine wrapper

Connection id:          16288
Current database:
Current user:           root@192.168.0.5
SSL:                    Not in use
Current pager:          stdout
Using outfile:          ''
Using delimiter:        ;
Server version:         5.6.34-log MySQL Community Server (GPL)
Protocol version:       10
Connection:             192.168.0.24 via TCP/IP
Server characterset:    utf8
Db     characterset:    utf8
Client characterset:    utf8
Conn.  characterset:    utf8
TCP port:               8635
Uptime:                 5 hours 5 min 34 sec

Threads: 2  Questions: 62118  Slow queries: 0  Opens: 70  Flush tables: 2  Open tables: 0  Queries per second avg: 3.388
--------------
```

Uptime indicates the running time of the DB instance. The command output shows that the DB instance has not been rebooted frequently. The client disconnection is not caused by DB instance reboots.

Step 4 Check parameters. If the values of **wait_timeout** and **interactive_timeout** are too small, the client automatically stops connections that timed out.

Step 5 Change the values of **wait_timeout** and **interactive_timeout** as required without the need of rebooting the DB instance.

Step 6 After about 10 minutes, run **show databases** to check whether the connection is normal.



```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
+--------------------+
3 rows in set (0.00 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
+--------------------+
3 rows in set (0.00 sec)

mysql> 
```

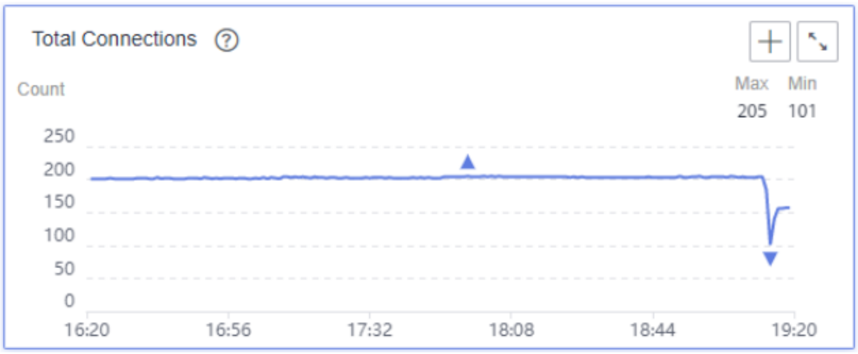
If information similar to the preceding figure is returned, the connection is normal.

----End

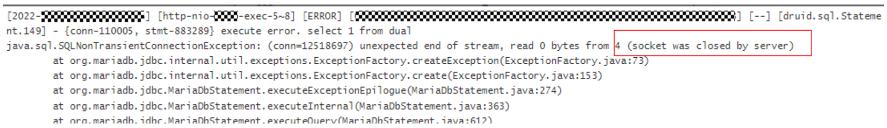
2.10 Disconnection Occurs Every 45 Days Due to the istio-citadel Certificate Mechanism

Scenario

The number of connections of multiple DB instances decreased sharply at the same time every 45 days. The following figure shows the number of total connections on the Cloud Eye console.



A large number of errors were reported on the client, as shown in the following figure.

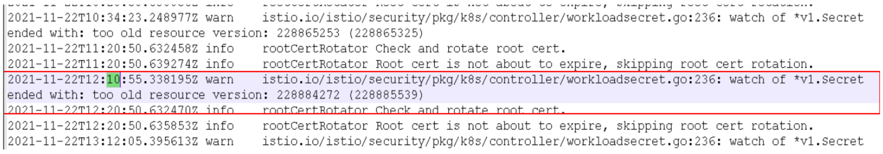


```
[2022-[redacted]] [http-nio-[redacted]-exec-5~8] [ERROR] [[redacted]] [--] [druid.sql.Stateme
nt.149] - {conn-110005, stmt-883289} execute error. select 1 from dual
java.sql.SQLNonTransientConnectionException: (conn=12518697) unexpected end of stream, read 0 bytes from 4 (socket was closed by server)
        at org.mariadb.jdbc.internal.util.exceptions.ExceptionFactory.createException(ExceptionFactory.java:73)
        at org.mariadb.jdbc.internal.util.exceptions.ExceptionFactory.create(ExceptionFactory.java:153)
        at org.mariadb.jdbc.MariaDbStatement.executeExceptionEpilogue(MariaDbStatement.java:274)
        at org.mariadb.jdbc.MariaDbStatement.executeInternal(MariaDbStatement.java:363)
        at org.mariadb.jdbc.MariaDbStatement.executeQuery(MariaDbStatement.java:612)
```

Possible Causes

1. Check whether a scheduled task with an interval of 45 days exists on the service side.

2. If the client uses a certificate encryption mechanism such as istio, analyze certificate-related logs and check whether information similar to the following is displayed: If yes, the problem is caused by certificate expiration.



```
2021-11-22T10:34:23.248977Z warn    istio.io/istio/security/pkg/k8s/controller/workloadsecret.go:236: watch of *v1.Secret
ended with: too old resource version: 228865253 (228865325)
2021-11-22T11:20:50.632458Z info    rootCertRotator Check and rotate root cert.
2021-11-22T11:20:50.639274Z info    rootCertRotator Root cert is not about to expire, skipping root cert rotation.
2021-11-22T12:10:55.338195Z warn    istio.io/istio/security/pkg/k8s/controller/workloadsecret.go:236: watch of *v1.Secret
ended with: too old resource version: 228884272 (228885539)
2021-11-22T12:20:50.632470Z info    rootCertRotator Check and rotate root cert.
2021-11-22T12:20:50.635853Z info    rootCertRotator Root cert is not about to expire, skipping root cert rotation.
2021-11-22T13:12:05.395613Z warn    istio.io/istio/security/pkg/k8s/controller/workloadsecret.go:236: watch of *v1.Secret
```

The expiration duration of the istio-citadel certificate is 45 days on the client. When the certificate has expired, the client initiates a database disconnection request.

Solution

- Set a proper expiration time for the istio-citadel certificate on the client and take preventive measures when the certificate expires.
- Check whether any other certificates have expired on the client.

3 SQL Issues

3.1 Invalid TIMESTAMP Default Value during Table Creation

Scenario

The CREATE TABLE statement failed to be executed.

```
CREATE TABLE cluster_membership
(
...
session_start TIMESTAMP DEFAULT '1970-01-01 00:00:01',
...
);
```

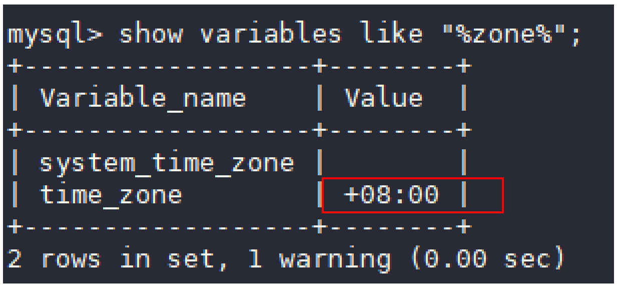
Failure cause: ERROR 1067: Invalid default value for 'session_start'

Possible Causes

The table column type is TIMESTAMP.

TaurusDB converts the value inserted to the TIMESTAMP column from the current time zone to the UTC time for storage. During query, it returns the value by converting the UTC time to the current time zone.

1. The time range for the TIMESTAMP column is from '1970-01-01 00:00:01' UTC to '2038-01-19 03:14:07' UTC. For details, see **the DATE, DATETIME, and TIMESTAMP types**.
2. Run the following command to check the time zone:
   ```
   show variables like "%zone%";
   ```
3. The UTC+8 time zone is used. The valid range for the default value starts from 1970-01-01 08:00:01.



```
mysql> show variables like "%zone%";
+------------------+--------+
| Variable_name    | Value  |
+------------------+--------+
| system_time_zone |        |
| time_zone        | +08:00 |
+------------------+--------+
2 rows in set, 1 warning (0.00 sec)
```

Solution

Change the default value of the TIMESTAMP column.

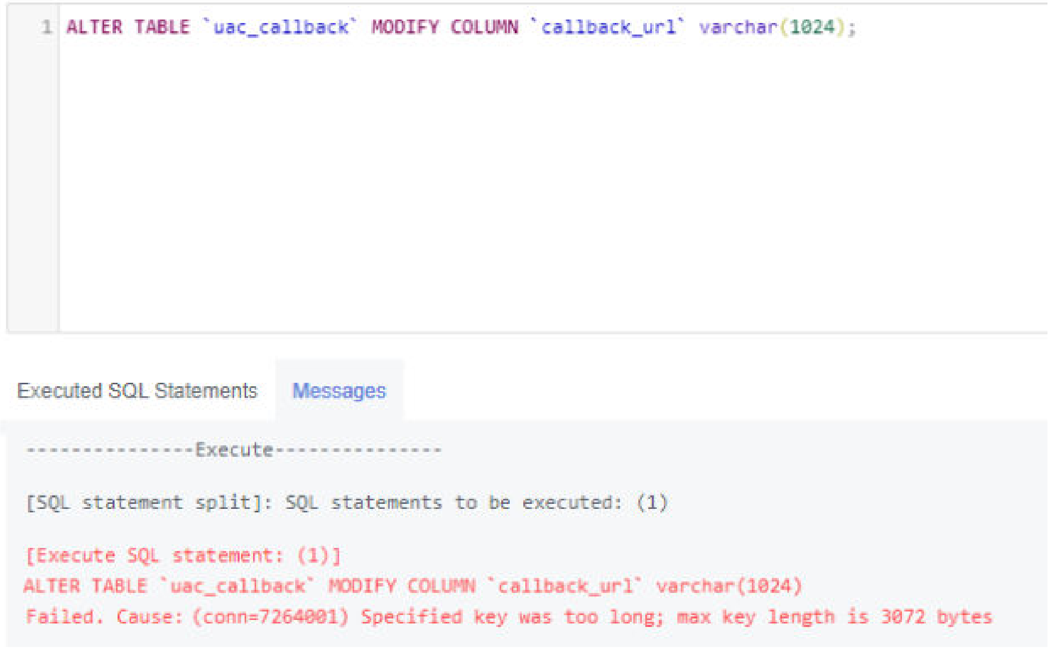
```
session_start TIMESTAMP DEFAULT '1970-01-01 08:00:01',
```

3.2 Failed to Change the VARCHAR Length Due to the Index Length Limit

Scenario

The **alter table** command failed to modify a table structure. The following error information was displayed:

```
Specified key was too long; max key length is 3072 bytes
```



```
ALTER TABLE `uac_callback` MODIFY COLUMN `callback_url` varchar(1024);
```

```
---------------Execute---------------

[SQL statement split]: SQL statements to be executed: (1)

[Execute SQL statement: (1)]
ALTER TABLE `uac_callback` MODIFY COLUMN `callback_url` varchar(1024)
Failed. Cause: (conn=7264001) Specified key was too long; max key length is 3072 bytes
```

Possible Causes

- If **innodb_large_prefix** is set to **OFF**, the allowed maximum length for a single-column index in an InnoDB table cannot exceed 767 bytes, while that for a composite index cannot exceed 3072 bytes, with each column in the composite index no more than 767 bytes.

- If **innodb_large_prefix** is set to **ON**, the allowed maximum length for a single-column index is 3072 bytes, and that for a composite index is also 3072 bytes.
- The index length is related to the character set. When the utf8 character set is used, a character occupies three bytes. If **innodb_large_prefix** is set to **ON**, the allowed maximum length for all columns in an index is 1072 characters.

The table structure is as follows:

```
CREATE TABLE `xxxxx` (
......
`subscription_type` varchar(64) NOT NULL DEFAULT 'DEVICE_EXCEPTION' COMMENT 'Subscription type',
`auth_key` varchar(255) DEFAULT'' COMMENT 'Signature. A token is added to the API request header based on the value of this parameter',
`create_time` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT 'Creation time',
`update_time` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP COMMENT 'Update time',
PRIMARY KEY (`id`) USING BTREE,
UNIQUE KEY `enterprise_id` (`subscription_type`,`enterprise_id`,`callback_url`) USING BTREE)
) ENGINE=InnoDB AUTO_INCREMENT=1039 DEFAULT CHARSET=utf8 ROW_FORMAT=DYNAMIC
```

This table uses the utf8 character set. One character occupies three bytes. The composite index **enterprise_id** contains the **callback_url** column. If a DDL operation is performed and **callback_url** to is changed to **varchar(1024)**, the maximum length of the composite index is exceeded. As a result, an error is reported.

Solution

Modify the index or column length.

3.3 Slow SQL Queries After a Large Amount of Data Is Deleted from a Large Table

Scenario

After multiple wide columns of data (the length of each record is about 1 GB) are deleted at a time, performing an INSERT, DELETE, UPDATE, or SELECT operation on the same table again takes an extended period of time. After about 20 minutes, the problem is resolved.

Problem Reproduction

1. Assume that the value of **max_allowed_packet** is **1073741824**.
2. Create a table.
```
CREATE TABLE IF NOT EXISTS zstest1
(
id int  PRIMARY KEY not null,
c_longtext LONGTEXT
);
```
3. Insert data to the table.
```
insert into zstest1 values(1, repeat('a', 1073741800));
insert into zstest1 values(2, repeat('a', 1073741800));
insert into zstest1 values(3, repeat('a', 1073741800));
insert into zstest1 values(4, repeat('a', 1073741800));
insert into zstest1 values(5, repeat('a', 1073741800));
insert into zstest1 values(6, repeat('a', 1073741800));
insert into zstest1 values(7, repeat('a', 1073741800));
insert into zstest1 values(8, repeat('a', 1073741800));
insert into zstest1 values(9, repeat('a', 1073741800));
insert into zstest1 values(10, repeat('a', 1073741800));
```

4. Delete data from the table.

```
delete from zstest1;
```

5. Execute a query statement.

```
select id from zstest1;    //The execution is slow.
```

Possible Causes

After the DELETE operation is performed, the background purge thread clears the records marked with delete mark. Due to the large amount of data to be deleted, the purge thread obtains the SX lock of the index root node where the page is located when traversing and releasing the page. As a result, the SELECT statement cannot obtain the RW lock of the root page and keeps waiting.

Solution

- This phenomenon is normal. After the purge operation is complete, the fault is rectified.
- Scale up the instance specifications to improve the purge efficiency.
- Do not delete a large amount of data at a time. To delete all data from a table, use the **truncate table** statement.

3.4 Error 1366 Reported When Data Containing Emojis Is Updated

Scenario

Error 1366 was reported when data containing emojis was inserted or updated.

```
java.sql.SQLException: Incorrect string value: '\xF0\x9F\x90\xB0\xE5\xA4...' for column 'username' at row 1 ;
uncategorized SQLException for SQL []; SQL state [HY000]; error code [1366];
Incorrect string value: '\xF0\x9F\x90\xB0\xE5\xA4...' for column 'username' at row 1;
```

Possible Causes

The cause is that the character set is incorrectly configured.

- An emoji is a special character and needs to be stored in a 4-byte character set.
- In this scenario, the database character set is utf-8, which supports a maximum of three bytes. The utf8mb4 character set supports a maximum of four bytes.

Solution

1. Change the character set for the field that stores emojis to utf8mb4.

   If a large number of tables and fields are involved, you are advised to set the encoding format of the tables and databases to utf8mb4. Sample commands:

   ALTER DATABASE database_name CHARACTER SET= utf8mb4 COLLATE= utf8mb4_unicode_ci;

ALTER TABLE table_name CONVERTTOCHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;

ALTER TABLE table_name MODIFY Field name VARCHAR(128) CHARSET utf8mb4 COLLATE utf8mb4_unicode_ci;

2. If the character set for the field is already utf8mb4, set the character sets of the client and server to utf8mb4.

3.5 Slow Stored Procedure Execution Due to Inconsistent Collations

Scenario

It took more than 1 minute to process a small amount of data using a stored procedure in a TaurusDB instance. However, executing the SQL statement in the stored procedure was much faster.

Possible Causes

The collation of the stored procedure is inconsistent with that of the related table and database. As a result, a large number of characters need to be converted in the query result, and the execution is slow.

Troubleshooting:

Run the following commands to view the definitions of the stored procedure and related table and check whether the collations are the same:

```
SHOW CREATE PROCEDURE xxx;
SHOW CREATE TABLE xxx
```

For example:

```
mysql> SHOW CREATE PROCEDURE testProc \G
*************************** 1. row ***************************
Procedure: showstuscore
sql_mode: STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION
Create Procedure: xxx
character_set_client: utf8mb4
collation_connection: utf8mb4_general_ci
Database Collation: utf8_general_ci
1 row in set (0.01 sec)
```

The collation of the stored procedure is **utf8mb4_general_ci**, but the collation of the database is **utf8_general_ci** by default. The collations are inconsistent, which may cause performance issues.

Solution

Change the collation of the stored procedure to be the same as that of the related table and database.

3.6 ERROR [1412] Reported for a DB Instance

Scenario

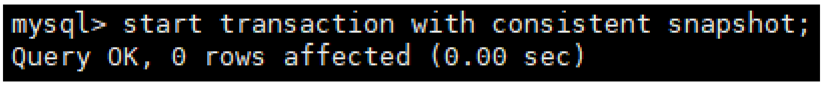
When an SQL statement was executed on a DB instance, the following error message was displayed:

```
ERROR[1412]:Table definition has changed, please retry transaction``
```

Possible Causes

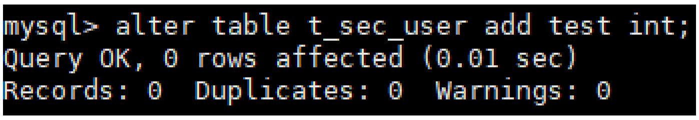
After a transaction with consistent snapshot was started, another session was executing DDL statements. Procedure for reproducing the problem:

1. Session 1 starts a transaction with consistent snapshot.



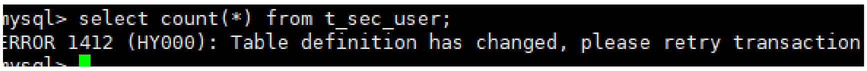
```
mysql> start transaction with consistent snapshot;
Query OK, 0 rows affected (0.00 sec)
```

2. Session 2 executes a DDL statement to modify the table structure.



```
mysql> alter table t_sec_user add test int;
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

3. Session 1 executes a common query statement.



```
mysql> select count(*) from t_sec_user;
ERROR 1412 (HY000): Table definition has changed, please retry transaction
mysql>
```

You can also analyze binlogs or audit logs to check whether a DDL statement and transaction with consistent snapshot are executed concurrently on the same table.

Solution

Do not execute a DDL statement and transaction with consistent snapshot concurrently on the same table.

3.7 Failed to Delete a Table with a Foreign Key

Scenario

When a table with a foreign key is deleted, the following error message will be displayed, which is irrelevant to user permissions:

```
ERROR 1451 (23000): Cannot delete or update parent row: a foreign key constraint fails ............
```

Possible Causes

A foreign key relationship exists between this table and another table. A link is established between the data in the two tables. To prevent foreign key constraints from being violated, data in the tables cannot be updated or deleted.

You can set **FOREIGN_KEY_CHECKS** to **off** to remove the foreign key relationship. For details, see **FOREIGN KEY Constraints**.

Solution

Set **FOREIGN_KEY_CHECKS** to **off**.

```
set session foreign_key_checks=off;
drop table table_name;
```

3.8 Incorrect GROUP_CONCAT Results

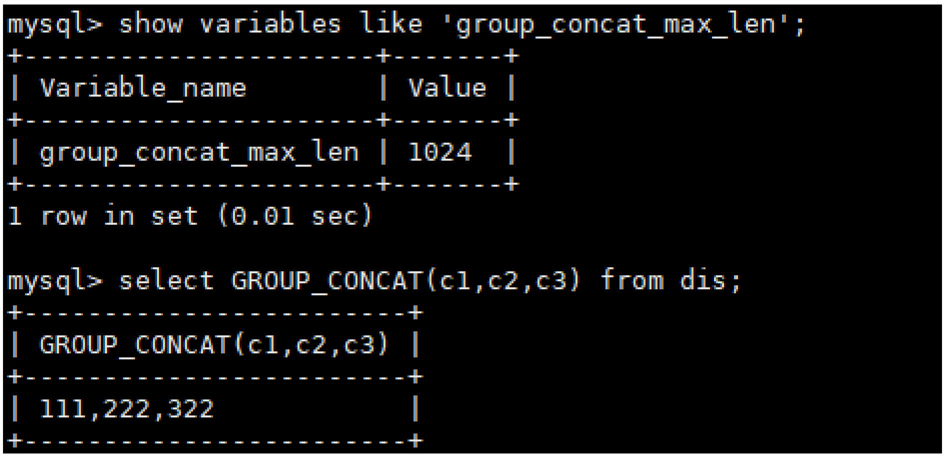
Scenario

When the GROUP_CONCAT() function was used in an SQL statement, the result did not meet the expectation.

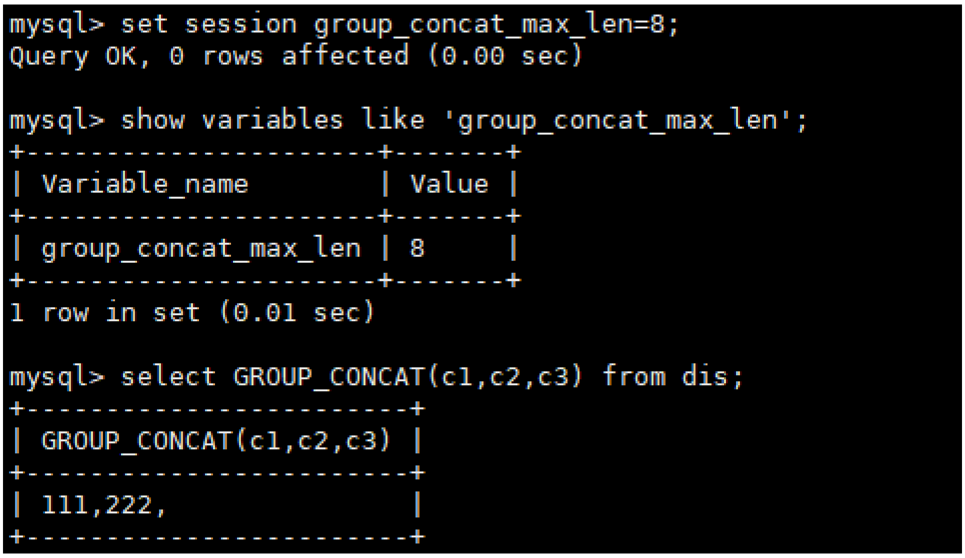
Possible Causes

The GROUP_CONCAT() function returned a string result consisting of concatenated values in the group. However, the **group_concat_max_len** parameter limited the result length of this function.

For example:



```
mysql> show variables like 'group_concat_max_len';
+----------------------+-------+
| Variable_name        | Value |
+----------------------+-------+
| group_concat_max_len | 1024  |
+----------------------+-------+
1 row in set (0.01 sec)

mysql> select GROUP_CONCAT(c1,c2,c3) from dis;
+------------------------+
| GROUP_CONCAT(c1,c2,c3) |
+------------------------+
| 111,222,322            |
+------------------------+
```



```
mysql> set session group_concat_max_len=8;
Query OK, 0 rows affected (0.00 sec)

mysql> show variables like 'group_concat_max_len';
+----------------------+-------+
| Variable_name        | Value |
+----------------------+-------+
| group_concat_max_len | 8     |
+----------------------+-------+
1 row in set (0.01 sec)

mysql> select GROUP_CONCAT(c1,c2,c3) from dis;
+------------------------+
| GROUP_CONCAT(c1,c2,c3) |
+------------------------+
| 111,222,               |
+------------------------+
```

Solution

Change the value of **group_concat_max_len** to adapt to the result length of the GROUP_CONCAT() function.

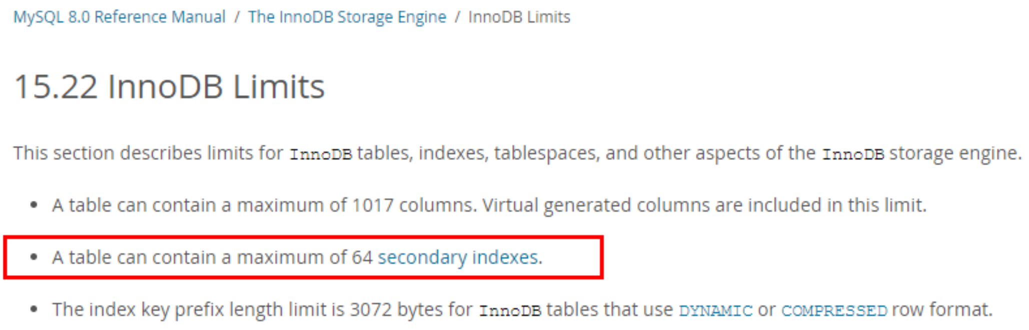
3.9 Error Message "Too many keys specified" Displayed When a Secondary Index Is Created

Scenario

A secondary index failed to be created, and the error message "Too many keys specified; max 64 keys allowed" was displayed.

Fault Analysis

TaurusDB limits the maximum number of secondary indexes in each InnoDB table to 64. If there are more than 64 secondary indexes, the error message "Too many keys specified; max 64 keys allowed" will be displayed. For details, see **InnoDB Limits**.



MySQL 8.0 Reference Manual / The InnoDB Storage Engine / InnoDB Limits

15.22 InnoDB Limits

This section describes limits for `InnoDB` tables, indexes, tablespaces, and other aspects of the `InnoDB` storage engine.

- A table can contain a maximum of 1017 columns. Virtual generated columns are included in this limit.
- A table can contain a maximum of 64 secondary indexes.
- The index key prefix length limit is 3072 bytes for `InnoDB` tables that use `DYNAMIC` or `COMPRESSED` row format.

Solution

Do not create too many indexes for a single table.

NOTE

Other restrictions on InnoDB tables:

1. A table can contain a maximum of 1017 columns (including virtual generated columns).
2. The index key prefix limit is 3072 bytes for InnoDB tables that use the DYNAMIC or COMPRESSED row format.
3. A maximum of 16 columns is permitted for multicolumn indexes. Exceeding the limit returns an error.

3.10 DISTINCT and GROUP BY Optimization

Scenario

The execution of the DISTINCT or GROUP BY statement is slow.

Possible Causes

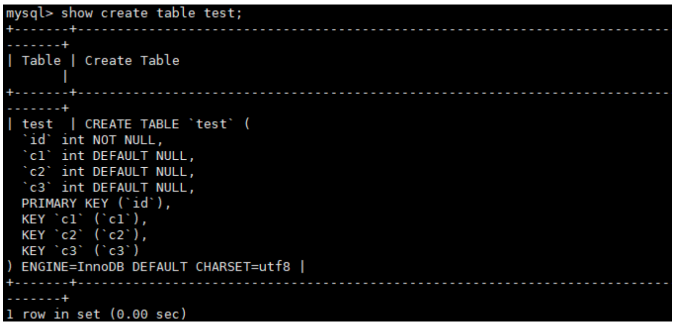
In most cases, DISTINCT can be converted into an equivalent GROUP BY statement. DISTINCT is mainly used to remove duplicate records from database tables and fetch only the unique records.

The DISTINCT statement groups data first, and then fetches a piece of data from each group and returns the data to the client. There are two scenarios for grouping data:

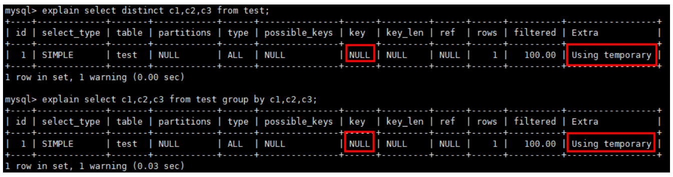
- All DISTINCT fields are included in the same index. In this scenario, TaurusDB directly uses the index to group data, obtains a piece of data from each group, and returns the data.
- Not all DISTINCT fields are included in the index. In this scenario, qualified data is written to a temporary table and grouped in the temporary table. Using temporary tables causes extra overhead, deteriorating the performance.

In conclusion, when using DISTINCT or GROUP BY, set an index that contains all dependent fields. The following is an optimization example:

- No proper index is available. As a result, temporary tables are used.

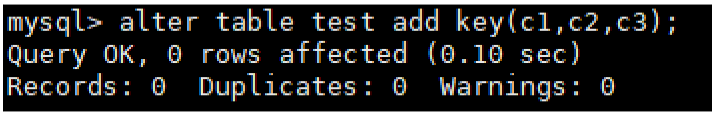


```
mysql> show create table test;
+-------+------------------------------------------------------------------------
-------+
| Table | Create Table
       |
+-------+------------------------------------------------------------------------
-------+
| test  | CREATE TABLE `test` (
  `id` int NOT NULL,
  `c1` int DEFAULT NULL,
  `c2` int DEFAULT NULL,
  `c3` int DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `c1` (`c1`),
  KEY `c2` (`c2`),
  KEY `c3` (`c3`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+-------+------------------------------------------------------------------------
-------+
1 row in set (0.00 sec)
```

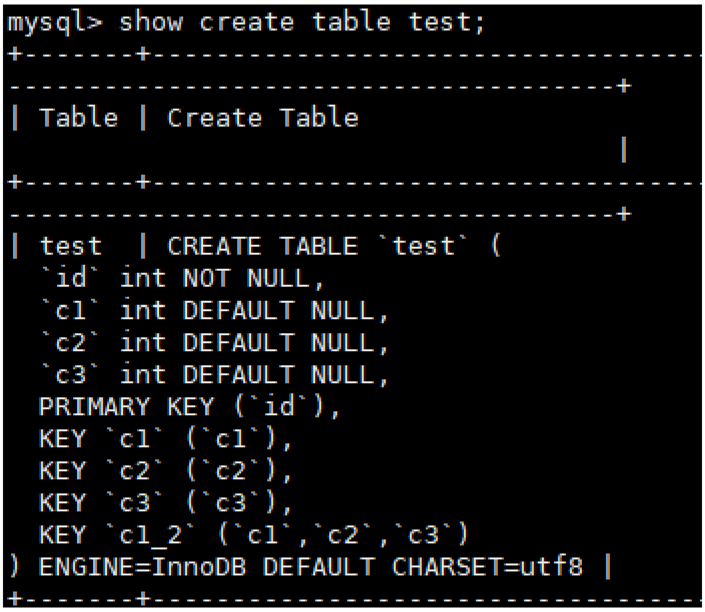


```
mysql> explain select distinct c1,c2,c3 from test;
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-----------------+
| id | select_type | table | partitions | type | possible_keys | key  | key_len | ref  | rows | filtered | Extra           |
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-----------------+
|  1 | SIMPLE      | test  | NULL       | ALL  | NULL          | NULL | NULL    | NULL |    1 |   100.00 | Using temporary |
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-----------------+
1 row in set, 1 warning (0.00 sec)

mysql> explain select c1,c2,c3 from test group by c1,c2,c3;
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-----------------+
| id | select_type | table | partitions | type | possible_keys | key  | key_len | ref  | rows | filtered | Extra           |
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-----------------+
|  1 | SIMPLE      | test  | NULL       | ALL  | NULL          | NULL | NULL    | NULL |    1 |   100.00 | Using temporary |
+----+-------------+-------+------------+------+---------------+------+---------+------+------+----------+-----------------+
1 row in set, 1 warning (0.03 sec)
```

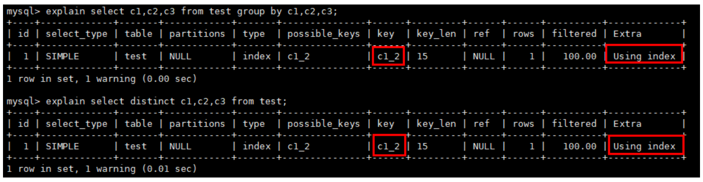
- A proper index is available, and temporary tables are not required.



```
mysql> alter table test add key(c1,c2,c3);
Query OK, 0 rows affected (0.10 sec)
Records: 0  Duplicates: 0  Warnings: 0
```



```
mysql> show create table test;
+-------+----------------------------------
--------------------------------------+
| Table | Create Table
                                      |
+-------+----------------------------------
--------------------------------------+
| test  | CREATE TABLE `test` (
  `id` int NOT NULL,
  `c1` int DEFAULT NULL,
  `c2` int DEFAULT NULL,
  `c3` int DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `c1` (`c1`),
  KEY `c2` (`c2`),
  KEY `c3` (`c3`),
  KEY `c1_2` (`c1`,`c2`,`c3`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+-------+----------------------------------
```



```
mysql> explain select c1,c2,c3 from test group by c1,c2,c3;
+----+-------------+-------+------------+-------+---------------+------+---------+------+------+----------+-------------+
| id | select_type | table | partitions | type  | possible_keys | key  | key_len | ref  | rows | filtered | Extra       |
+----+-------------+-------+------------+-------+---------------+------+---------+------+------+----------+-------------+
|  1 | SIMPLE      | test  | NULL       | index | c1_2          | c1_2 | 15      | NULL |    1 |   100.00 | Using index |
+----+-------------+-------+------------+-------+---------------+------+---------+------+------+----------+-------------+
1 row in set, 1 warning (0.00 sec)

mysql> explain select distinct c1,c2,c3 from test;
+----+-------------+-------+------------+-------+---------------+------+---------+------+------+----------+-------------+
| id | select_type | table | partitions | type  | possible_keys | key  | key_len | ref  | rows | filtered | Extra       |
+----+-------------+-------+------------+-------+---------------+------+---------+------+------+----------+-------------+
|  1 | SIMPLE      | test  | NULL       | index | c1_2          | c1_2 | 15      | NULL |    1 |   100.00 | Using index |
+----+-------------+-------+------------+-------+---------------+------+---------+------+------+----------+-------------+
1 row in set, 1 warning (0.01 sec)
```

Solution

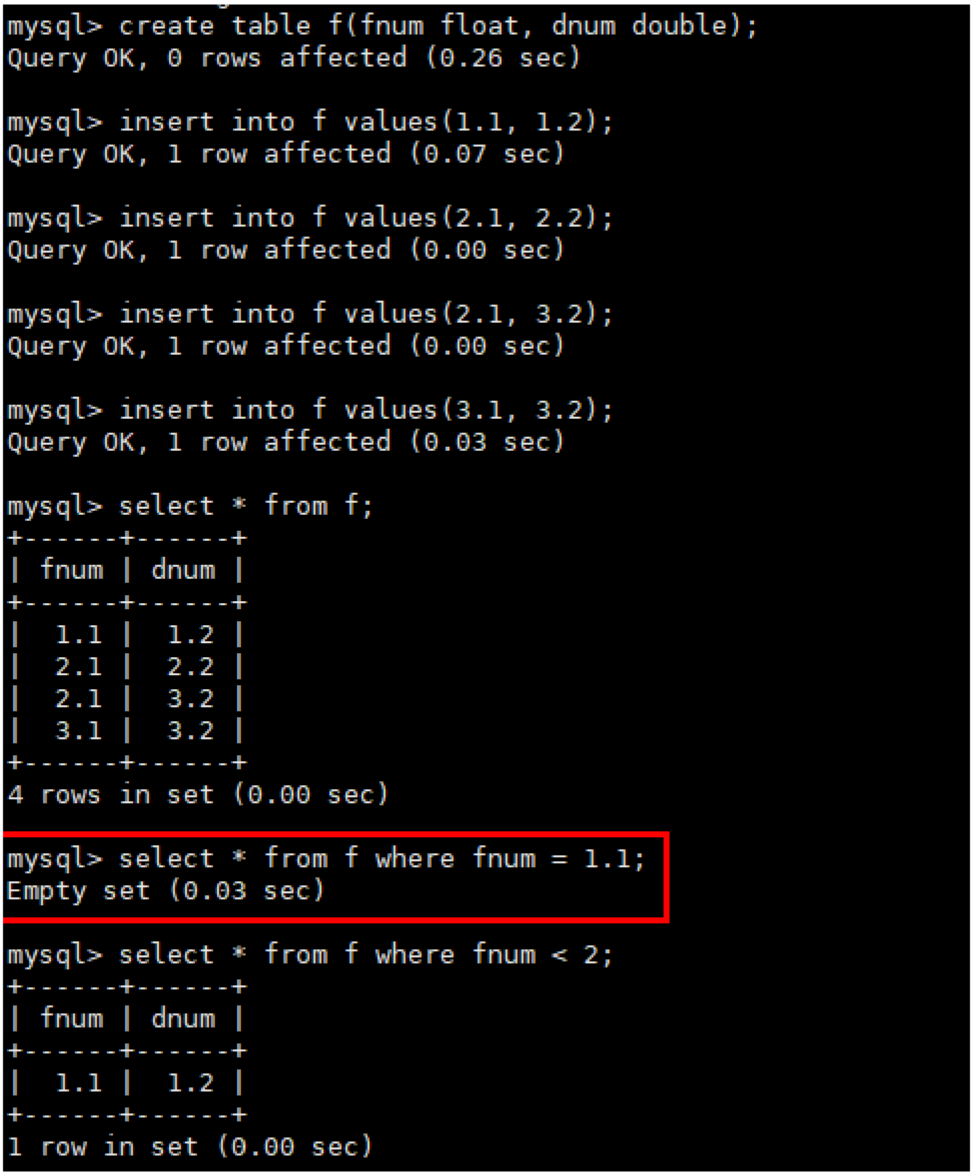
When using DISTINCT or GROUP BY, create an index that contains all dependent fields.

3.11 Equivalent Comparison Failures with Floating-Point Numbers

Possible Causes

Equivalent comparison of floating-point numbers is a common problem. In computing, floating-point numbers are stored as approximate values instead of exact values. Therefore, unexpected problems may occur during equivalent comparison and mathematical operations.

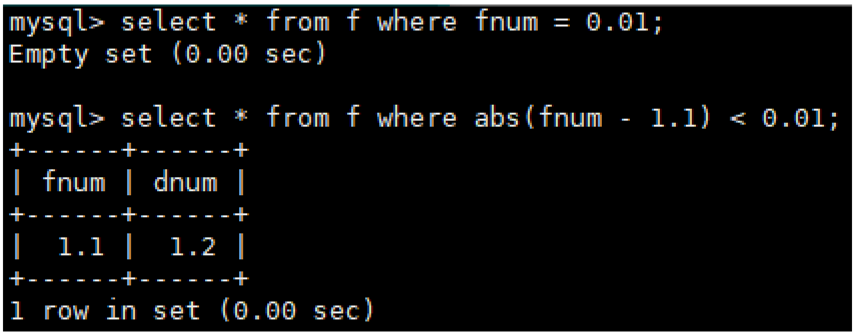
In TaurusDB, FLOAT and DOUBLE are floating-point types. The following figure shows an example for equivalent comparison using floating-point numbers:



```
mysql> create table f(fnum float, dnum double);
Query OK, 0 rows affected (0.26 sec)

mysql> insert into f values(1.1, 1.2);
Query OK, 1 row affected (0.07 sec)

mysql> insert into f values(2.1, 2.2);
Query OK, 1 row affected (0.00 sec)

mysql> insert into f values(2.1, 3.2);
Query OK, 1 row affected (0.00 sec)

mysql> insert into f values(3.1, 3.2);
Query OK, 1 row affected (0.03 sec)

mysql> select * from f;
+------+------+
| fnum | dnum |
+------+------+
|  1.1 |  1.2 |
|  2.1 |  2.2 |
|  2.1 |  3.2 |
|  3.1 |  3.2 |
+------+------+
4 rows in set (0.00 sec)

mysql> select * from f where fnum = 1.1;
Empty set (0.03 sec)

mysql> select * from f where fnum < 2;
+------+------+
| fnum | dnum |
+------+------+
|  1.1 |  1.2 |
+------+------+
1 row in set (0.00 sec)
```

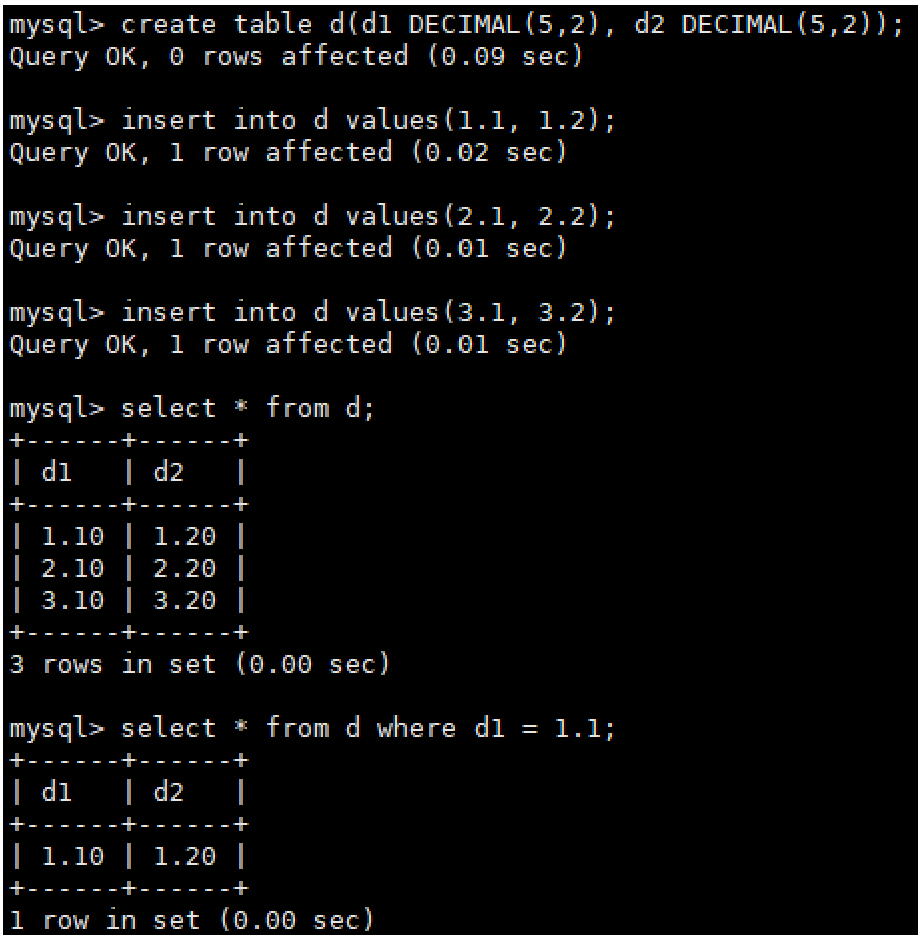
Solution

1. Decide on an acceptable tolerance for differences between the field and the value and then do the comparison against the tolerance value. For example:



```
mysql> select * from f where fnum = 0.01;
Empty set (0.00 sec)

mysql> select * from f where abs(fnum - 1.1) < 0.01;
+------+------+
| fnum | dnum |
+------+------+
|  1.1 |  1.2 |
+------+------+
1 row in set (0.00 sec)
```

2. Use the fixed-point number type (DECIMAL) to replace the floating-point number type. Example:



```
mysql> create table d(d1 DECIMAL(5,2), d2 DECIMAL(5,2));
Query OK, 0 rows affected (0.09 sec)

mysql> insert into d values(1.1, 1.2);
Query OK, 1 row affected (0.02 sec)

mysql> insert into d values(2.1, 2.2);
Query OK, 1 row affected (0.01 sec)

mysql> insert into d values(3.1, 3.2);
Query OK, 1 row affected (0.01 sec)

mysql> select * from d;
+------+------+
| d1   | d2   |
+------+------+
| 1.10 | 1.20 |
| 2.10 | 2.20 |
| 3.10 | 3.20 |
+------+------+
3 rows in set (0.00 sec)

mysql> select * from d where d1 = 1.1;
+------+------+
| d1   | d2   |
+------+------+
| 1.10 | 1.20 |
+------+------+
1 row in set (0.00 sec)
```

3.12 A Large Number of SELECT Requests Routed to The Primary Instance After Database Proxy Is Enabled

Possible Causes

1. Read weight parameter

   This parameter specifies read weights distributed to the primary node and read replicas. It is only applied when there are read replicas.

   For example, if a DB instance contains one primary node and two read replicas and the read weights are set, respectively, to 1, 2, and 3 for the primary node and two read replicas, read requests are distributed to the primary node and read replicas based on the ratio of 1:2:3. If the read weights are set to 0, 2, and 3, respectively, read requests are distributed to only the read replicas based on the ratio of 2:3.

   For more information, see **Assigning Read Weights**.

2. Transactions

   SQL statements in a transaction are sent to the primary instance. If **set autocommit=0** is configured before a query statement is executed, the query statement is routed to the primary node as a transaction.

3. Connection binding

If multi-statements (for example, **insert xxx;select xxx**) are executed, all subsequent requests will be routed to the primary node because the SQL statement for creating temporary tables binds the connection to the primary node. To restore read/write splitting, disconnect your application from your instance and then connect it back again.

4. Custom variables

   SQL statements containing custom variables will be routed to the primary node.

5. Read operations with locks (for example, **SELECT for UPDATE**) will be routed to the primary node.
6. You can use hints to specify whether an SQL statement is routed to the primary node or read replicas
   - **/*FORCE_MASTER*/**: An SQL statement is executed on the primary node.
   - **/*FORCE_SLAVE*/**: An SQL statement is executed on read replicas.

   Hints are only used as routing suggestions. In non-read-only SQL and non-transaction scenarios, SQL statements cannot be routed to read replicas.

7. Session consistency

   In a given session, read requests may be sent to the primary node.

   For more information, see **Consistency Levels**.

3.13 Tablespace Bloat

Scenario

Tablespace bloat often occurs in TaurusDB instances. For example, a table contains only 11,774 rows of data but occupies 49.9 GB of storage space. After the table is exported to a local directory, it occupies only 800 MB.

Possible Causes

Cause 1: Parallel Migration During DRS Full Migration

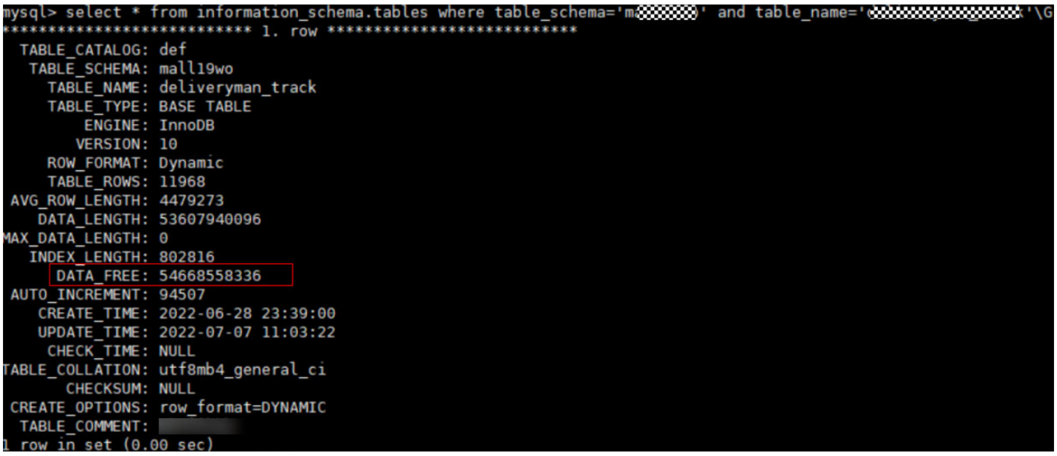
During full migration, DRS uses row-level parallel migration to ensure migration performance and transmission stability. If the source database data is compact, tablespace bloat may occur after data is migrated to the TaurusDB database. As a result, the disk space required is much greater than that of the source database.

Cause 2: Table Fragmentation After a Large Number of Deletions Are Performed

When data is deleted, TaurusDB does not reclaim the storage occupied by the deleted data. Instead, it only marks the deletion and fills the space with new data if any. If there is no data to fill up the space, tablespace bloat occurs, along with table fragmentation.

You can run the following SQL statement to query detailed information about a table. The **DATA_FREE** field indicates the size of tablespace fragmentation.

```
select * from information_schema.tables where table_schema='db_name' and table_name = 'table_name'\G
```



```
mysql> select * from information_schema.tables where table_schema='m[redacted]' and table_name='[redacted]'\G
*************************** 1. row ***************************
  TABLE_CATALOG: def
   TABLE_SCHEMA: mall19wo
     TABLE_NAME: deliveryman_track
     TABLE_TYPE: BASE TABLE
         ENGINE: InnoDB
        VERSION: 10
     ROW_FORMAT: Dynamic
     TABLE_ROWS: 11968
 AVG_ROW_LENGTH: 4479273
    DATA_LENGTH: 53607940096
MAX_DATA_LENGTH: 0
   INDEX_LENGTH: 802816
      DATA_FREE: 54668558336
 AUTO_INCREMENT: 94507
    CREATE_TIME: 2022-06-28 23:39:00
    UPDATE_TIME: 2022-07-07 11:03:22
     CHECK_TIME: NULL
TABLE_COLLATION: utf8mb4_general_ci
       CHECKSUM: NULL
 CREATE_OPTIONS: row_format=DYNAMIC
  TABLE_COMMENT: [redacted]
1 row in set (0.00 sec)
```

Solution

Run the following SQL command to optimize the tablespace:

```
optimize table table_name;
```

NOTE

The **optimize table** command locks tables for a short period of time. Therefore, you are advised to optimize tablespaces during off-peak hours.

3.14 ERROR 1396 Reported When a User Is Created

Scenario

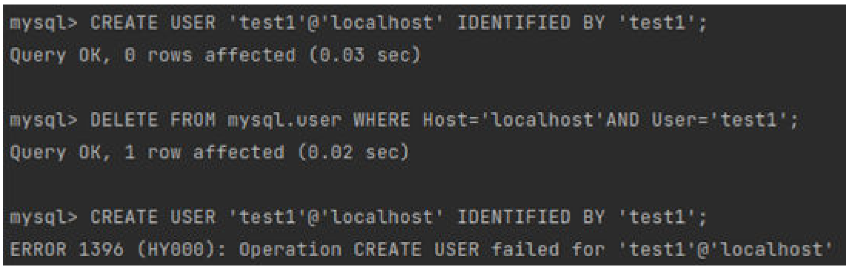
A user account disappeared from the console, but the account and its password could still be used to connect to the instance.

When a new account with the same name as the disappeared account was created, the following error information was displayed:

```
ERROR 1396 (HY000): Operation CREATE USER failed for xxx.
```

Possible Causes

1. The disappeared account has been deleted from the **mysql.user** table and therefore was not displayed on the console.
2. Because the account and its password could still be used to log in to the instance, the account was deleted using **delete from mysql.user**. If you use **delete from mysql.user** to delete an account, you also need to run the **flush privileges** command to delete related data from the memory. Then, the account cannot log in to the instance any more.
3. The reason why a new account with the same name as the disappeared account could not be created is that there was still related data about the disappeared account in the memory.



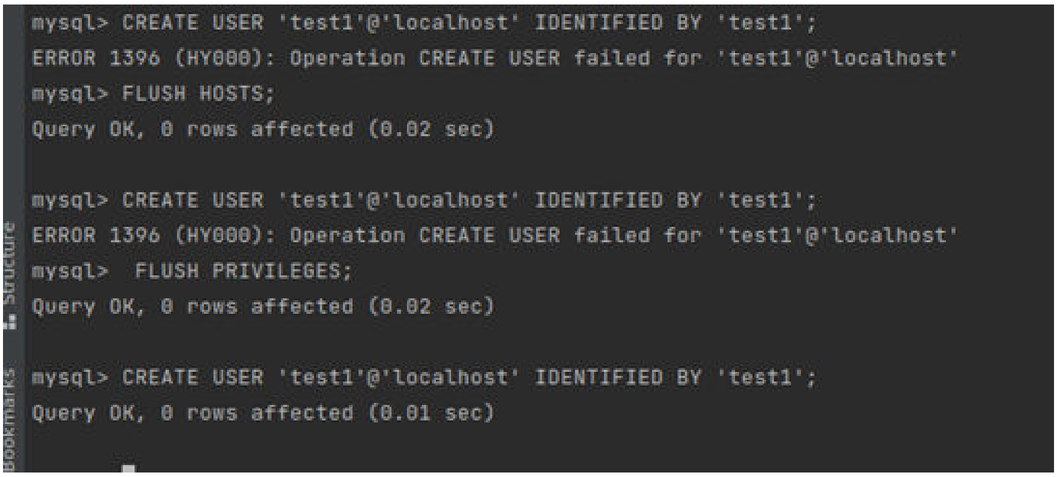
```
mysql> CREATE USER 'test1'@'localhost' IDENTIFIED BY 'test1';
Query OK, 0 rows affected (0.03 sec)

mysql> DELETE FROM mysql.user WHERE Host='localhost'AND User='test1';
Query OK, 1 row affected (0.02 sec)

mysql> CREATE USER 'test1'@'localhost' IDENTIFIED BY 'test1';
ERROR 1396 (HY000): Operation CREATE USER failed for 'test1'@'localhost'
```

The correct way to delete an account is using the **drop user** statement. When running **drop user**, pay attention to that:

- **drop user** can be used to delete one or more users and revoke their permissions.
- **drop user** requires the DELETE permission or the global CREATE USER permission on the database.
- If the host name of the account is not specified in the **drop user** statement, the host name **%** is used by default.

Troubleshooting example:

After an account is created, the **delete** statement is used to delete the account. When a new account with the same name as the disappeared account is created, error 1396 is reported. After the **flush privileges** command is executed, an account with the same name can be created.



```
mysql> CREATE USER 'test1'@'localhost' IDENTIFIED BY 'test1';
ERROR 1396 (HY000): Operation CREATE USER failed for 'test1'@'localhost'
mysql> FLUSH HOSTS;
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE USER 'test1'@'localhost' IDENTIFIED BY 'test1';
ERROR 1396 (HY000): Operation CREATE USER failed for 'test1'@'localhost'
mysql>  FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE USER 'test1'@'localhost' IDENTIFIED BY 'test1';
Query OK, 0 rows affected (0.01 sec)
```

Solution

- Method 1 (recommended): During off-peak hours, run the **drop user user_name** command as the administrator to delete the disappeared account and then create an account with the same name.
- Method 2: During off-peak hours, run the **flush privileges** command as the administrator and then create an account with the same name. You are advised to enable SQL Explorer to locate which client deletes the user.

3.15 Error Message Reported When alter table xxx discard/import tablespace Is Executed

Scenario

When **alter table xxx discard** or **import tablespace** is executed in TaurusDB, the following error message was displayed: "ERROR 3658 (HY000): Feature IMPORT/DISCARD TABLESPACE is unsupported ()".

Possible Causes

MySQL Community Edition uses **alter table xxx discard** or **import tablespace** to physically replace table data based on local **.ibd** tablespace files for data migration, data backups, and data restoration.

TaurusDB decouples storage from compute. Data is stored using shared storage and there is no local **.ibd** file.

Solution

Import and export data, use DRS for data synchronization, or back up data and restore data.

3.16 Native Error 1461 Reported by a DB Instance

Scenario

The following error information is displayed when there are large amounts of concurrent read and write requests, large amounts of SQL statements, or in data migration scenarios:

mysql_stmt_prepare failed! error(1461)Can't create more than max_prepared_stmt_count statements (current value: 16382)

Fault Analysis

The **max_prepared_stmt_count** value ranges from **0** to **1048576**. The default value is **16382**. This parameter limits the total number of prepared statements in all sessions on mysqld. The current value exceeds the value range of this parameter.

Solution

Set **max_prepared_stmt_count** to a larger value. The recommended value is **65535**.

3.17 "Row size too large" Reported When a Table Failed to Be Created

Scenario

A table failed to be created and the following information is displayed:

Row size too large. The maximum row size for the used table type, not counting BLOBs, is 65535. This includes storage overhead, check the manual. You have to change some columns to TEXT or BLOBs

Fault Analysis

The total length of the **varchar** fields exceeds 65535, resulting in a table creation failure.

Solution

1. Reduce the length.
   ```
   CREATE TABLE t1 (a VARCHAR(10000),b VARCHAR(10000),c VARCHAR(10000),d VARCHAR(10000),e
   VARCHAR(10000),f VARCHAR(10000) ) ENGINE=MyISAM  CHARACTER SET latin1;
   ```
2. Change a column to **TEXT** by referring to the **Limits on Table Column Count and Row Size**.

3.18 Duplicate Data Exists After ORDER BY LIMIT Is Executed

Scenario

Sorting a table and paginating the results did not yield the expected outcome.

Suppose there is a table called **merchants** with only two columns: **id** and **category**. The table structure is as follows:

```
mysql> show create table merchants;
+-----------
+------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------+
| Table     | Create
Table
                                          |
+-----------
+------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------+
| merchants | CREATE TABLE `merchants` (
  `id` int NOT NULL AUTO_INCREMENT,
  `category` int DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=11 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci |
+-----------
+------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

Run the following SQL statement to view data in the table:

```
mysql>  select * from merchants;
+----+----------+
| id | category |
+----+----------+
|  1 |        1 |
|  2 |        3 |
|  3 |        2 |
|  4 |        2 |
|  5 |        1 |
|  6 |        2 |
|  7 |        3 |
|  8 |        3 |
|  9 |        2 |
| 10 |        1 |
+----+----------+
10 rows in set (0.00 sec)
```

Run the following SQL statement to sort the data by **category**:

```
mysql> select * from merchants order by category;
+----+----------+
| id | category |
+----+----------+
|  1 |        1 |
|  5 |        1 |
| 10 |        1 |
|  3 |        2 |
|  4 |        2 |
|  6 |        2 |
|  9 |        2 |
|  2 |        3 |
|  7 |        3 |
|  8 |        3 |
+----+----------+
10 rows in set (0.00 sec)
```

Run the following SQL statement to paginate the sorted results with a limit of two rows per page:

```
mysql> select * from merchants order by category limit 0,2;
+----+----------+
| id | category |
+----+----------+
|  1 |        1 |
|  5 |        1 |
+----+----------+
2 rows in set (0.00 sec)

mysql> select * from merchants order by category limit 2,2;
+----+----------+
| id | category |
+----+----------+
|  1 |        1 |
|  9 |        2 |
+----+----------+
2 rows in set (0.00 sec)
```

The data on the second page is incorrect. According to the original sorting order without pagination, the second page should display rows with IDs 10 and 3. However, the actual result shows rows with IDs 1 and 9.

Possible Causes

When the optimizer encounters an ORDER BY LIMIT statement, the optimizer uses a priority queue structure for sorting. However, this sorting method is considered unstable. Once the LIMIT *n* results are filtered, they are directly returned without any guarantee of order.

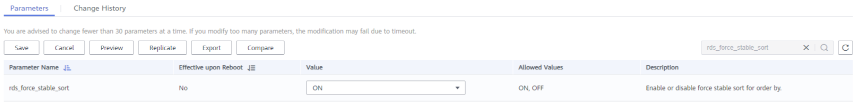
Solution

- Solution 1: Add an index to the column that needs to be sorted.

  Example: **alter table ratings add index idx_category (category);**
- Solution 2: Add the primary key column after ORDER BY in the sorting statement.

  Example: **select * from ratings order by category, id limit 2,2;**
- Solution 3: On the **Parameters** page of the TaurusDB console, set **rds_force_stable_sort** to **ON**. This parameter forces the use of stable sorting algorithms to ensure stable sorting results.

Figure 3-1 Setting the **rds_force_stable_sort** parameter



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.

Parameter Name	Effective upon Reboot	Value	Allowed Values	Description
rds_force_stable_sort	No	ON	ON, OFF	Enable or disable force stable sort for order by.

3.19 Error Message Reported When select * from sys.innodb_lock_waits Is Executed

Scenario

The output of the following statement was not as expected.

select * from sys.innodb_lock_waits

The following error message was displayed:

```
Execution failed. Cause: (conn=14690848) Illegal mix of collations (utf8mb4_0900_ai_ci,IMPLICIT) and
(utf8mb4_general_ci,IMPLICIT) for operation '='
```

Possible Causes

This error usually occurs when two tables, fields, or databases with different character set encodings are compared.

Solution

If the character set used by the client for connection is different from that used by the database, adjust the configuration of the client to let it use the same character set to connect to the database, or explicitly set the character set of the client after it is connected to the database.

You can explicitly set the character set as follows:

set collation_connection = utf8mb4_general_ci;

utf8mb4_general_ci indicates the desired character set and collation rule.

If the following information is displayed, the character set is set successfully:

```
Query OK, 0 rows affected (0.00 sec)
```

4 Parameter-related Issues

4.1 Global Parameters Failed to Be Modified on the Client

Scenario

A global parameter failed to be modified on the client, the error message "ERROR 1227 (42000): Access denied" was displayed.

Possible Causes

TaurusDB does not support global parameter modifications using commands.

Solution

Log in to the console and modify the parameters on the console.

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** > **TaurusDB**.

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

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

----End

4.2 Connection Exit Due to Improper Timeout Parameter Settings

Scenario

A database connection exit often occurs. As a result, subsequent statements fail to be executed.

Possible Causes

When a connector or API is used to connect to a database, the client has some default parameter settings. The settings of some important parameters, such as **socketTimeout** and **connectTimeout**, determine the client connection timeout duration. If the wait time of a connection exceeds the value of one of these parameters, the connection will be interrupted.

Solution

- Change the default values of parameters such as **socketTimeout** and **connectTimeout** to appropriate values.
- Pay attention to the reconnection function in the program.
- **Using connection pools is recommended.**

4.3 long_query_time Changes Fail to Take Effect

Scenario

The value of **long_query_time** was successfully changed on the console, but changed value failed to be applied.

Possible Causes

When you change the **long_query_time** value on the console, the system actually uses **set global** *<variable name>* **=** *<new variable value>* to modify global parameters.

The new parameter value cannot be applied for the current connection and other connections that have been connected to the database. It means that the new parameter value is applied only for new connections. After you disconnect and reconnect all connections, the new parameter value is applied.

Example

Commands in this example explain how to apply the changed parameter value.

1. Create session 1.

```
Check the value of long_query_time.
show variables like 'long_query_time';
+-----------------+-----------+
| Variable_name   | Value     |
+-----------------+-----------+
| long_query_time | 10.000000 |
+-----------------+-----------+
1 row in set (0.08 sec)
Change the value of long_query_time.
set global long_query_time=1;
Query OK, 0 rows affected (0.02 sec)
# View the value of long_query_time. The changed value is not applied.
show variables like 'long_query_time';
+-----------------+-----------+
| Variable_name   | Value     |
+-----------------+-----------+
| long_query_time | 10.000000 |
+-----------------+-----------+
1 row in set (0.01 sec)
```

2. Create session 2.

```
show variables like 'long_query_time';
+-----------------+-----------+
| Variable_name   | Value     |
+-----------------+-----------+
| long_query_time | 10.000000 |
+-----------------+-----------+
1 row in set (0.01 sec)
```

3. Execute the following commands in session 1.

```
# After the set global command is executed in connection 1, the changed parameter value is not
applied.
show variables like 'long_query_time';
+-----------------+-----------+
| Variable_name   | Value     |
+-----------------+-----------+
| long_query_time | 10.000000 |
+-----------------+-----------+
1 row in set (0.01 sec)
# After you disconnect session 1 and reconnect it, the new parameter value is applied.
show variables like 'long_query_time';
+-----------------+----------+
| Variable_name   | Value    |
+-----------------+----------+
| long_query_time | 1.000000 |
+-----------------+----------+
1 row in set (0.00 sec)
```

4. Disconnect session 2 and reconnect it. The new parameter value is applied.

```
show variables like 'long_query_time';
+-----------------+----------+
| Variable_name   | Value    |
+-----------------+----------+
| long_query_time | 1.000000 |
+-----------------+----------+
1 row in set (0.01 sec)
```

4.4 Timeout Parameters

The following table lists the TaurusDB timeout parameters.

Table 4-1 Parameter description

Parameter	Reboot Required	Description
connect_timeout	No	Number of seconds that the TaurusDB server waits for a connection packet before responding with **Bad handshake**. If the network quality is poor, you can increase the value of this parameter.
innodb_flush_log_at_timeout	No	How frequently the log buffer flushes to disk, in seconds. This parameter is valid only when **innodb_flush_log_at_trx_commit** is set to **2**.
innodb_lock_wait_timeout	No	Length of time in seconds an InnoDB transaction waits for a row lock before giving up.
parallel_queue_timeout	No	Waiting time for the parallel execution. If the number of parallel-executed threads in the system is still greater than the value of **parallel_max_threads** after the waiting time, new queries will be executed in single-thread mode.
lock_wait_timeout	No	Timeout in seconds for attempts to acquire metadata locks
net_read_timeout	No	Number of seconds to wait for more data from a connection before aborting the read
net_write_timeout	No	Number of seconds to wait for a block to be written to a connection before aborting the write
interactive_timeout	No	Number of seconds the server waits for activity on an interactive connection before closing it
wait_timeout	No	Number of seconds the server waits for activity on a non-interactive connection before closing it

5 Performance Issues

5.1 High CPU Usage

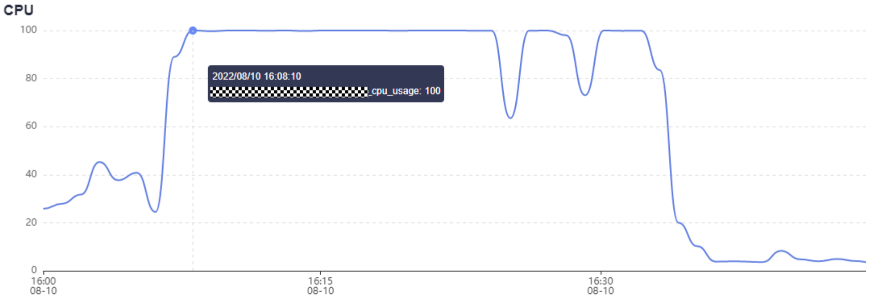
Scenario

The SQL statement execution of a TaurusDB instance slowed down at about 16:08, and a timeout error was reported.

Troubleshooting

1. Check the CPU usage. In this example, the CPU usage of the instance soared to 100% at about 16:08 and remained at the high line.

Figure 5-1 CPU usage



2. Check the QPS, slow SQL queries, and active connections. The QPS and active connections increased sharply at about 16:08 and a large number of slow SQL queries were generated.

Figure 5-2 QPS

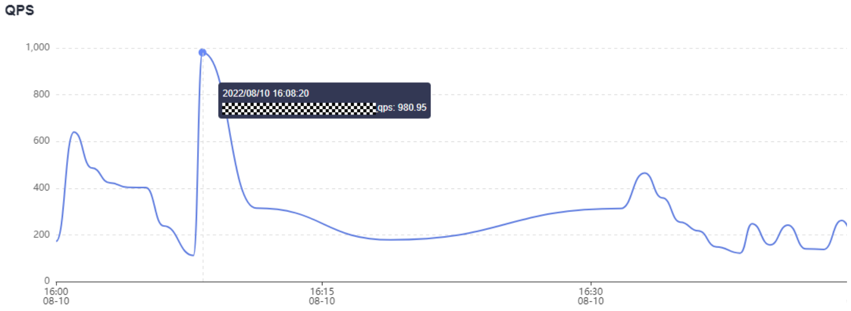


Figure 5-3 Active connections

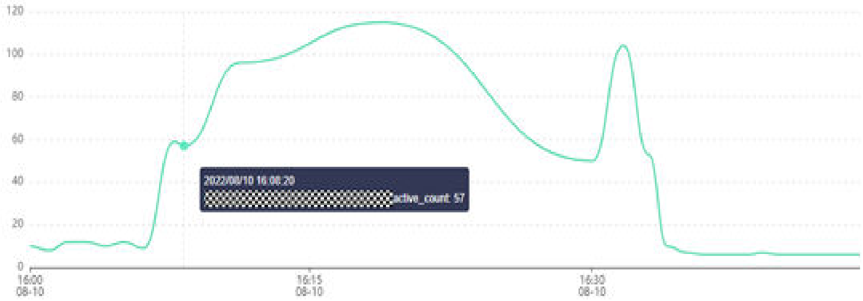
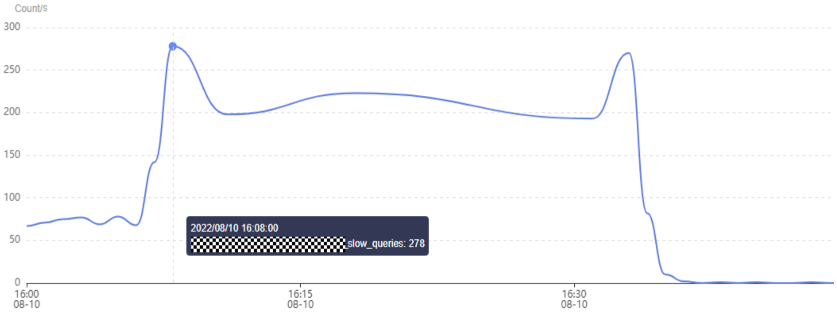
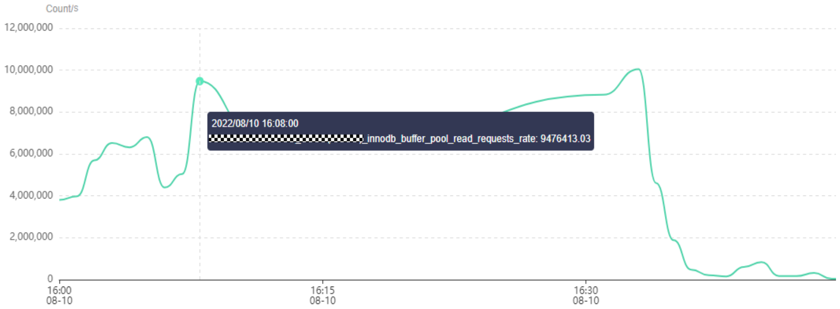


Figure 5-4 Slow SQL queries

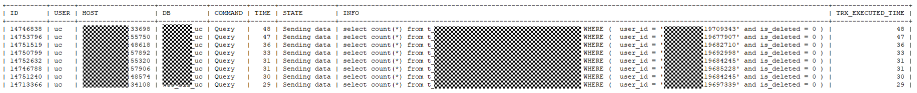


3. Check the InnoDB logical read rate. The logical read rate of InnoDB also increased sharply around 16:08, and the pattern was similar to that for the slow SQL queries.

Figure 5-5 InnoDB logical read rate

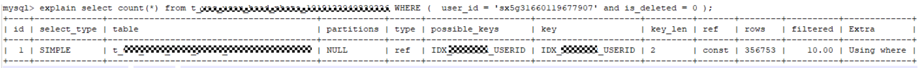


4. Log in to the instance and check real-time sessions. There were a large number of sessions executing **SELECT COUNT(*)**.



```
+----------+------+-------------------+-----------+---------+------+--------------+------------------------------------------------------------------------------------------------------------------+--------------------+
| ID       | USER | HOST              | DB        | COMMAND | TIME | STATE        | INFO                                                                                                             | TRX_EXECUTED_TIME |
+----------+------+-------------------+-----------+---------+------+--------------+------------------------------------------------------------------------------------------------------------------+--------------------+
| 14746838 | uc   | [redacted]:33698 | [redacted]_uc | Query |   48 | Sending data | select count(*) from t_[redacted] WHERE (  user_id = '[redacted]19709343' and is_deleted = 0 ) |                 48 |
| 14753796 | uc   | [redacted]:55750 | [redacted]_uc | Query |   47 | Sending data | select count(*) from t_[redacted] WHERE (  user_id = '[redacted]19677907' and is_deleted = 0 ) |                 47 |
| 14751519 | uc   | [redacted]:48618 | [redacted]_uc | Query |   36 | Sending data | select count(*) from t_[redacted] WHERE (  user_id = '[redacted]19682710' and is_deleted = 0 ) |                 36 |
| 14750799 | uc   | [redacted]:57892 | [redacted]_uc | Query |   33 | Sending data | select count(*) from t_[redacted] WHERE (  user_id = '[redacted]19692998' and is_deleted = 0 ) |                 33 |
| 14752632 | uc   | [redacted]:55320 | [redacted]_uc | Query |   31 | Sending data | select count(*) from t_[redacted] WHERE (  user_id = '[redacted]19684245' and is_deleted = 0 ) |                 31 |
| 14746788 | uc   | [redacted]:57906 | [redacted]_uc | Query |   31 | Sending data | select count(*) from t_[redacted] WHERE (  user_id = '[redacted]19685228' and is_deleted = 0 ) |                 31 |
| 14751240 | uc   | [redacted]:48574 | [redacted]_uc | Query |   30 | Sending data | select count(*) from t_[redacted] WHERE (  user_id = '[redacted]19684245' and is_deleted = 0 ) |                 30 |
| 14713366 | uc   | [redacted]:34108 | [redacted]_uc | Query |   29 | Sending data | select count(*) from t_[redacted] WHERE (  user_id = '[redacted]19697339' and is_deleted = 0 ) |                 29 |
```

Run **EXPLAIN** to check the execution plan of the SQL statement. It was found that the SQL statement was not indexed and the entire table was scanned with rows to be scanned reaching over 350,000.



```
mysql> explain select count(*) from t_[redacted] WHERE (  user_id = 'sx5g31660119677907' and is_deleted = 0 );
+----+-------------+-----------------------------------------+------------+------+---------------------+---------------------+---------+-------+--------+----------+-------------+
| id | select_type | table                                   | partitions | type | possible_keys       | key                 | key_len | ref   | rows   | filtered | Extra       |
+----+-------------+-----------------------------------------+------------+------+---------------------+---------------------+---------+-------+--------+----------+-------------+
|  1 | SIMPLE      | t_[redacted]                            | NULL       | ref  | IDX[redacted]_USERID | IDX_[redacted]_USERID | 2       | const | 356753 |    10.00 | Using where |
+----+-------------+-----------------------------------------+------------+------+---------------------+---------------------+---------+-------+--------+----------+-------------+
```

5. Check the table structure. Only the IDX_XX_USERID index was added for the **is_deleted** field. Therefore, no index was available for the preceding query. After an index was added for the **idx_user_id** field, the CPU usage of the instance decreased to a normal level at about 16:37 and services were recovered.

Solution

1. Before deploying new workloads, use **EXPLAIN** and SQL diagnosis tools to analyze the execution plans of key SQL statements and add indexes based on the optimization suggestions to avoid full table scanning.
2. If the high CPU usage is caused by a large number of concurrent requests, upgrade the DB instance specifications or use exclusive resources to avoid CPU contention, or create read replicas to reduce the read pressure of the primary instance.
3. Use **show processlist** to view the current session information to locate the fault. Any session whose status is **Sending data**, **Copying to tmp table**, **Copying to tmp table on disk**, **Sorting result**, or **Using filesort** may have performance problems.
4. In emergency scenarios, enable SQL statement concurrency control or kill sessions to temporarily limit the number of slow SQL queries.

5.2 Out of Memory (OOM) Errors

TaurusDB Memory Description

The memory of a TaurusDB instance can be roughly divided into two parts: globally shared memory and session-level private memory.

- Shared memory is allocated upon the creation of an instance based on parameter settings and is shared by all connections.
- Private memory is allocated by the system upon connection to the TaurusDB instance and is released only when the connection is released.

Inefficient SQL statements or improper database parameter settings may increase memory usage and even cause an OOM error during peak hours.

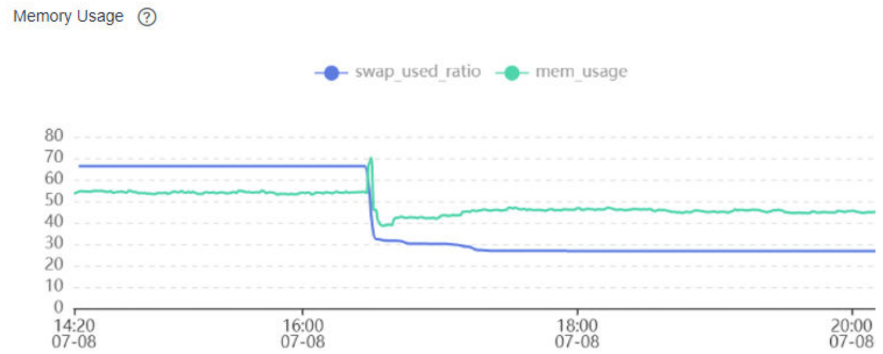
Scenario

The memory usage of a TaurusDB instance increased sharply at 16:30. An OOM error occurred and then the instance rebooted.

Troubleshooting

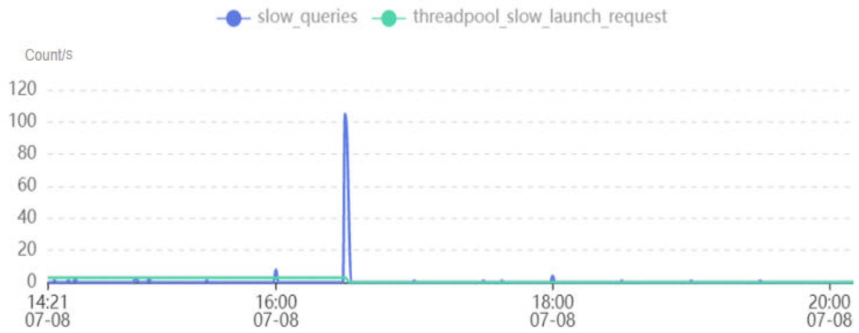
1. Check the memory usage. In this example, it shot up around 16:30.

Figure 5-6 Memory usage



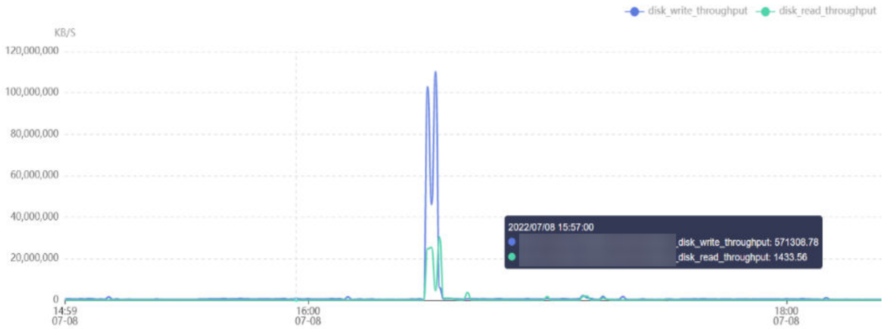
2. Check for slow SQL queries. The number of slow SQL queries increased sharply in that period.

Figure 5-7 Slow SQL queries



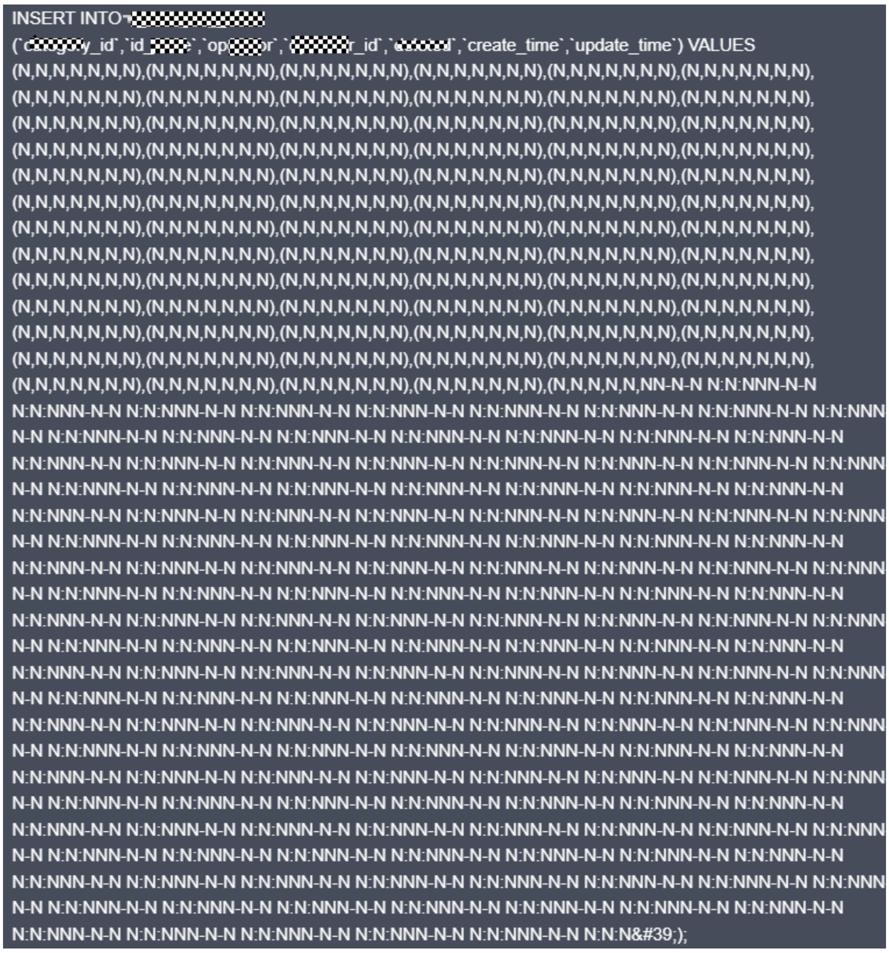
3. Check the disk throughput. There were a large number of read and write operations being performed on the disk in that period.

Figure 5-8 Disk throughput



4. Analyze slow query logs generated in that period. There were a large number of multi-value INSERT statements, which cause every session to request a large amount of session-level memory at the same time. Therefore, an OOM error occurred.

Figure 5-9 Slow query logs



```
INSERT INTO¬[redacted]
(`[redacted]y_id`,`id_[redacted]`,`op[redacted]r`,`[redacted]r_id`,`[redacted]`,`create_time`,`update_time`) VALUES
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),
(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,N,N),(N,N,N,N,N,NN-N-N N:N:NNN-N-N
N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN
N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N
N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN
N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N
N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN
N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N
N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN
N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N
N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN
N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N
N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN
N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N
N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN
N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N
N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN
N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N
N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN
N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N
N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN
N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N
N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:NNN-N-N N:N:N&#39;);
```

Solution

1. For the OOM error caused by multi-value INSERT statements, reduce the amount of data inserted at a time and disconnect sessions to release memory. You can run the **show full processlist** command to check whether there are sessions with high memory usage.
2. Set the session-level memory parameter to an appropriate value. You can estimate the maximum memory based on the following formula: Global memory + Session-level memory x Maximum number of sessions. Note that setting **performance_schema** to **ON** also causes memory overhead.
3. Upgrade the instance specifications to maintain the memory usage within a proper range, preventing a sudden increase in traffic from causing an OOM crash.

5.3 Tablespace Bloat

Scenario

Tablespace bloat often occurs in TaurusDB instances. For example, a table contains only 11,774 rows of data but occupies 49.9 GB of storage space. After the table is exported to a local directory, it occupies only 800 MB.

Possible Causes

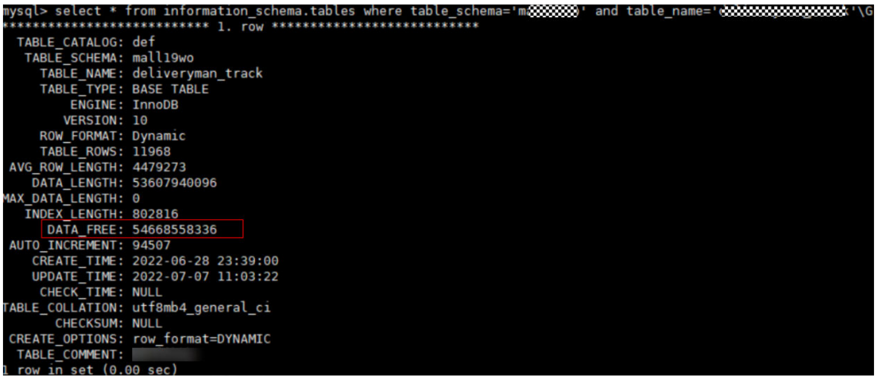
Cause 1: Parallel Migration During DRS Full Migration

During full migration, DRS uses row-level parallel migration to ensure migration performance and transmission stability. If the source database data is compact, tablespace bloat may occur after data is migrated to the TaurusDB database. As a result, the disk space required is much greater than that of the source database.

Cause 2: Table Fragmentation After a Large Number of Deletions Are Performed

When data is deleted, TaurusDB does not reclaim the storage occupied by the deleted data. Instead, it only marks the deletion and fills the space with new data if there is any. If there is no data to fill up the space, tablespace bloat occurs, along with table fragmentation.

You can run the following SQL statements to query details about a table. The **DATA_FREE** field indicates the size of tablespace fragmentation.

1. Updating statistics
   ```
   analyze table db_name.table_name;
   ```
2. Checking the fragmentation size
   ```
   select * from information_schema.tables where table_schema='db_name' and table_name = 'table_name'\G;
   ```
   
   ```
   mysql> select * from information_schema.tables where table_schema='m[redacted]' and table_name='[redacted]'\G
   *************************** 1. row ***************************
     TABLE_CATALOG: def
      TABLE_SCHEMA: mall19wo
        TABLE_NAME: deliveryman_track
        TABLE_TYPE: BASE TABLE
            ENGINE: InnoDB
           VERSION: 10
        ROW_FORMAT: Dynamic
        TABLE_ROWS: 11968
    AVG_ROW_LENGTH: 4479273
       DATA_LENGTH: 53607940096
   MAX_DATA_LENGTH: 0
      INDEX_LENGTH: 802816
         DATA_FREE: 54668558336
    AUTO_INCREMENT: 94507
       CREATE_TIME: 2022-06-28 23:39:00
       UPDATE_TIME: 2022-07-07 11:03:22
        CHECK_TIME: NULL
   TABLE_COLLATION: utf8mb4_general_ci
          CHECKSUM: NULL
    CREATE_OPTIONS: row_format=DYNAMIC
     TABLE_COMMENT: [redacted]
   1 row in set (0.00 sec)
   ```

Solution

Run the following SQL command to optimize the tablespace:

```
optimize table table_name;
```

The **optimize table** command briefly locks tables, so you should optimize tablespaces during off-peak hours.

5.4 Read Replica Uses Far More Storage Than the Primary Node

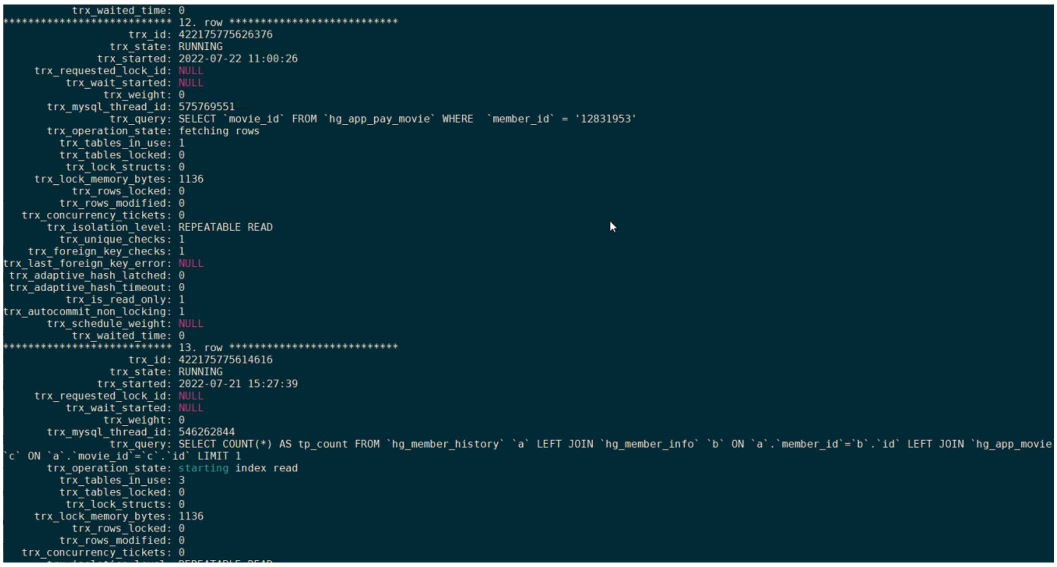
Scenario

The storage usage of a TaurusDB read replica was, for example, 195 GB higher than that of the primary node.

Troubleshooting

The storage space is calculated as follows: Shared storage space + Space occupied by binlogs + Space occupied by data disks (for storing temporary tables)

Check the transactions running on the read replica.



```
           trx_waited_time: 0
*************************** 12. row ***************************
                    trx_id: 422175775626376
                 trx_state: RUNNING
               trx_started: 2022-07-22 11:00:26
     trx_requested_lock_id: NULL
          trx_wait_started: NULL
                trx_weight: 0
       trx_mysql_thread_id: 575769551
                 trx_query: SELECT `movie_id` FROM `hg_app_pay_movie` WHERE  `member_id` = '12831953'
       trx_operation_state: fetching rows
         trx_tables_in_use: 1
         trx_tables_locked: 0
          trx_lock_structs: 0
     trx_lock_memory_bytes: 1136
           trx_rows_locked: 0
         trx_rows_modified: 0
   trx_concurrency_tickets: 0
       trx_isolation_level: REPEATABLE READ
         trx_unique_checks: 1
    trx_foreign_key_checks: 1
trx_last_foreign_key_error: NULL
 trx_adaptive_hash_latched: 0
 trx_adaptive_hash_timeout: 0
          trx_is_read_only: 1
trx_autocommit_non_locking: 1
       trx_schedule_weight: NULL
           trx_waited_time: 0
*************************** 13. row ***************************
                    trx_id: 422175775614616
                 trx_state: RUNNING
               trx_started: 2022-07-21 15:27:39
     trx_requested_lock_id: NULL
          trx_wait_started: NULL
                trx_weight: 0
       trx_mysql_thread_id: 546262844
                 trx_query: SELECT COUNT(*) AS tp_count FROM `hg_member_history` `a` LEFT JOIN `hg_member_info` `b` ON `a`.`member_id`=`b`.`id` LEFT JOIN `hg_app_movie
`c` ON `a`.`movie_id`=`c`.`id` LIMIT 1
       trx_operation_state: starting index read
         trx_tables_in_use: 3
         trx_tables_locked: 0
          trx_lock_structs: 0
     trx_lock_memory_bytes: 1136
           trx_rows_locked: 0
         trx_rows_modified: 0
   trx_concurrency_tickets: 0
```

As shown in the preceding figure, there was a long transaction started a day ago that has not been committed yet. The temporary tables generated by this transaction were not cleared, which took up a lot of storage space.

Solution

- Method 1: Wait until the transaction is committed. After that, the temporary tables will be automatically cleared, releasing storage space.
- Method 2: Kill the corresponding session to stop the long transaction.

5.5 Slow SQL Execution Due to Hot and Cold Data Problems

Scenario

When you migrate data from a self-managed MySQL database or a peer vendor's MySQL database to a TaurusDB instance on the cloud, the execution speed of an SQL statement is much lower than that of the source database.

Possible Causes

The execution speed of an SQL statement differs greatly when it is executed for the first time and the second time. This is determined by the MySQL buffer pool mechanism.

- When the statement is executed for the first time, data is stored on the disk, which is called cold data. Reading cold data takes a certain period of time.

- The data you have queried is then cached in the buffer pool of the memory. It is called hot data and can be quickly accessed in the memory. When you execute the statement for the second time, data is read from the buffer pool, which is much faster than reading data from disks.

In this troubleshooting case, the data you queried in the source database is frequently accessed data, that is, hot data. Thus, it can be read at a high speed. After the data is migrated to the TaurusDB instance, when you execute the SQL statement on the new database for the first time, the data you expect to query is probably cold data. This time, the access speed is slow. If you run the statement again, the data access speed will greatly improve.

Solution

This issue is not an exception. In a database, it usually takes much time to execute a statement for the first time, but when the statement is executed again, it gets much faster. The access speed improves because reading hot data from the buffer pool is much faster than reading cold data from disks.

5.6 Full Storage Caused by Complex Queries

Scenario

The storage usage of the primary node or read replica is occasionally high or reaches 100%, while the storage usage of other read replicas is within a normal range.

Possible Causes

When you run complex queries on data of a TaurusDB database, TaurusDB creates temporary tables to store the data and operations such as GROUP BY, ORDER BY, DISTINCT, and UNION are executed on the data in the temporary tables. When memory is insufficient, storage space is consumed.

Troubleshooting:

1. Check the storage usage of other read replicas. If the storage usage of such read replicas is normal, the high storage usage of the primary node or read replica is related to SQL queries running on it.
2. Check the instance slow query logs to find whether any slow-running queries occurred when the storage usage was high.
3. If there is a slow-running query, run the **explain** *[slow SQL statement]* command to analyze the SQL statement.
4. Check whether the **extra** column in the command output contains **using temporary** or **using filesort**. If yes, a temporary table or file is used during the statement execution. If a large amount of data is queried, the storage usage is high.

Solution

1. Optimize the query statement by adopting the following measures:

   - Add a proper index.
   - Use the WHERE condition.
   - Rewrite the SQL statement to optimize the execution plan.
   - If temporary tables are necessary, reduce the number of concurrent requests.

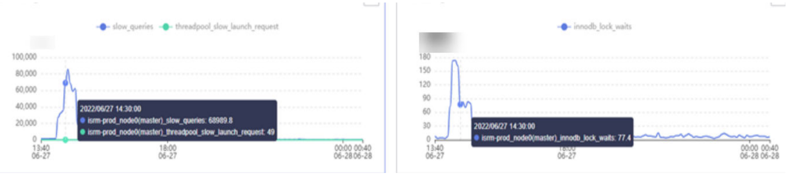
2. Workaround: Scale up storage space temporarily. Optimizing complex query statements cannot reduce the storage usage right away.

5.7 Slow Response Due to Deadlocks

Scenario

A large number of row lock conflicts occurred in a database between 14:00 and 15:00. The database response became slow because a large number of update and insert sessions in the kernel were waiting for row lock release and the CPU usage reached about 70%.

The following figure shows the row lock waits and metadata locks on the Cloud Eye console.



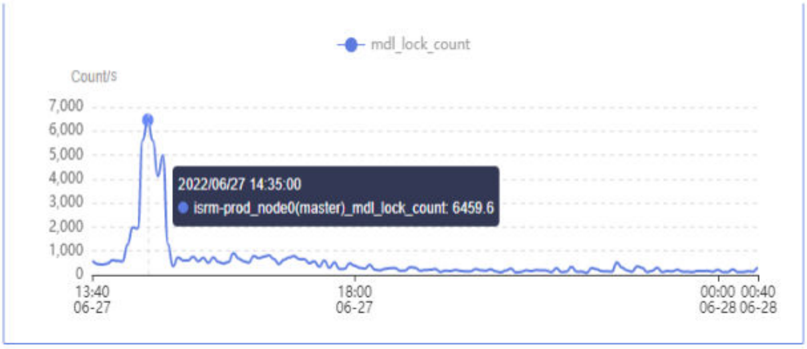


Table where a deadlock occurred:

```
********* 1. row *********
Table: table_test Create Table: CREATE TABLE table_test(
...
CONSTRAINT act_fk_exe_parent FOREIGN KEY (parent_id_) REFERENCES act_ru_execution (id_) ON DELETE
CASCADE,
CONSTRAINT act_fk_exe_procdef FOREIGN KEY (proc_def_id_) REFERENCES act_re_procdef (id_),
CONSTRAINT act_fk_exe_procinst FOREIGN KEY (proc_inst_id_) REFERENCES act_ru_execution (id_) ON
DELETE CASCADE ON UPDATE CASCADE, CONSTRAINT act_fk_exe_super FOREIGN KEY (super_exec_)
REFERENCES act_ru_execution (id_) ON DELETE CASCADE ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_bin
```

Possible Causes

1. Deadlocks occurred in some tables. As a result, the CPU usage increased.

2. If a table contains a large number of foreign keys, updating records in the table requires not only the row lock of the table but also the corresponding locks of the tables associated with its foreign keys. In high concurrency scenarios, lock conflicts or deadlocks are more likely to occur than common tables. For details, see **FOREIGN KEY Constraints**.
3. When detecting a deadlocked table, TaurusDB rolls back the transaction. The tables associated with the foreign keys of the deadlocked table are also impacted. As a result, the database response becomes slow.

Solution

Check and optimize deadlocked tables and use proper foreign keys to avoid update conflicts and deadlocks.

5.8 CPU Usage Increase

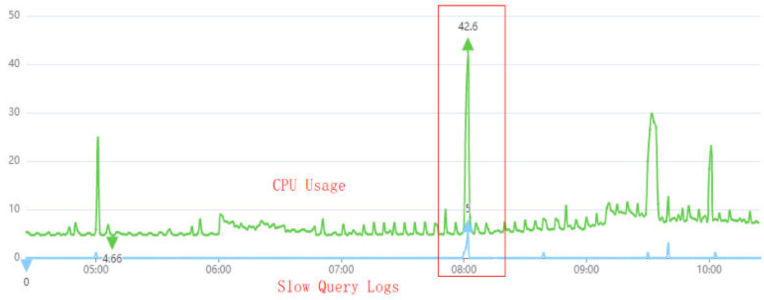
If the CPU usage of your TaurusDB instance increases or reaches 100%, the database response may become slow and new connections may time out.

Scenario 1: CPU Usage Increase Caused by Slow Queries

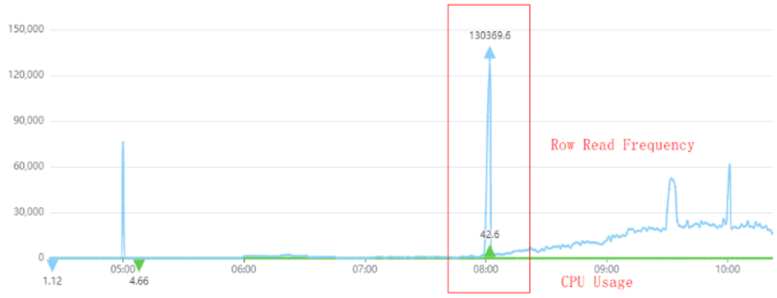
Cause: A large number of slow SQL queries cause an increase in CPU usage. The slow SQL queries need to be optimized.

Troubleshooting:

View the CPU usage and slow query logs.



- If a large number of slow query logs are generated and the number change is consistent with the CPU usage curve, the CPU usage increase is caused by slow SQL queries.
- If there are a few slow query logs but the number change is basically consistent with the CPU usage curve, check whether the row read rate change is consistent with the CPU curve.



If yes, the CPU usage increase is caused by access to a large amount of row data. Although there are a small number of slow SQL queries, the queries need to access a large amount of row data, causing high average I/O. Therefore, even if the QPS is not high (for example, the website access traffic is not heavy), the CPU usage of the instance is also high.

Solution:

1. View slow query logs generated within the corresponding time period.
2. Pay attention to slow queries with more than one million rows scanned or more than one million rows returned, and slow queries with long lock waiting time.
3. Analyze slow queries or use **SQL Diagnosis**.
4. Create read replicas and enable Database Proxy to split read and write requests. Read replicas can offload the read pressure from the primary instance, thus improving the database throughput. For details, see **What Is Database Proxy**.
5. Analyze live sessions on the database to locate slow SQL statements.
    a. Connect to the database.
    b. Run the **show full processlist;** command.
    c. Analyze sessions that take a long time to execute and are in the **Sending data**, **Copying to tmp table**, **Copying to tmp table on disk**, **Sorting result**, or **Using filesort** state.

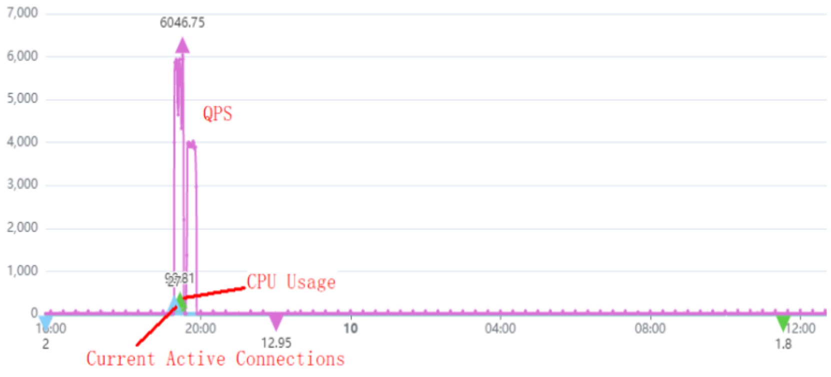
Scenario 2: CPU Usage Increase Caused by Increased Connections and QPS

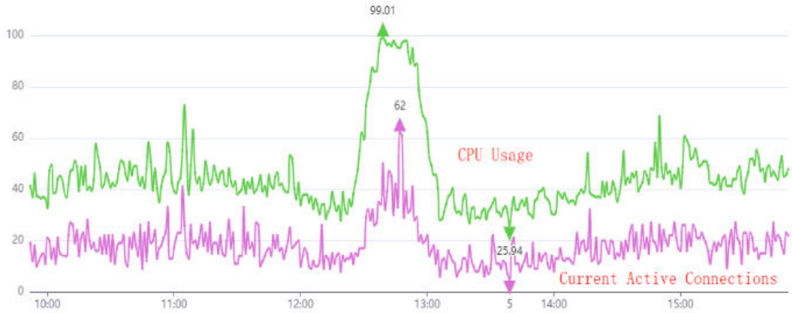
Cause: Increased requests cause an increase in CPU usage.

Troubleshooting:

Check whether the changes of the QPS, active connections, total connections, and CPU usage are consistent.

QPS refers to the number of queries per second. If the QPS and active connections increase at the same time, and the QPS curve matches the CPU usage curve, the CPU usage increase is caused by increased requests, as shown in the following figure.





In this scenario, SQL statements are usually simple and the execution efficiency is high. There is little room for optimization on SQL statements. You need to optimize the database.

Solution:

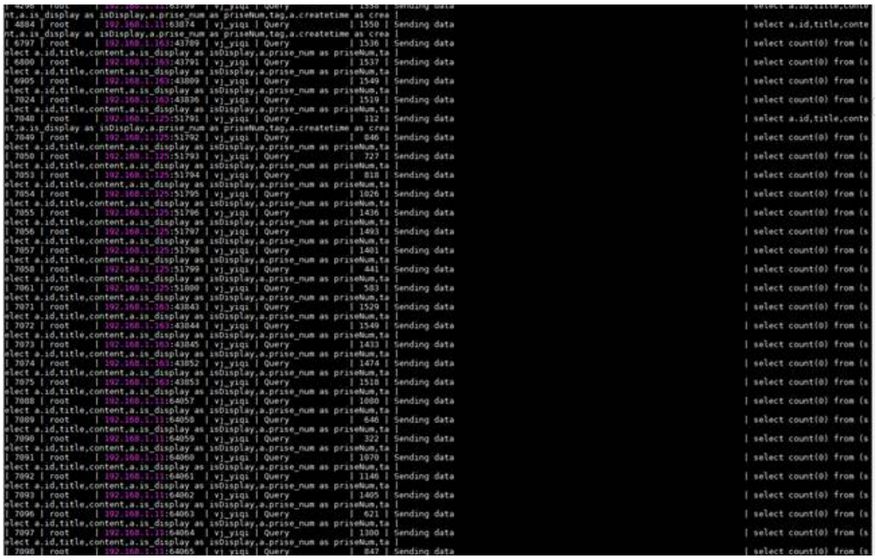
1. Upgrade the vCPU specifications of your instance because this problem usually occurs in instances with smaller vCPU specifications.
2. Optimize slow queries by referring to **Scenario 1: CPU Usage Increase Caused by Slow Queries**. If this method is not so helpful, upgrade the vCPU specifications of your instance.
3. Use database and table sharding for tables with a large amount of data to reduce the amount of data accessed in a single query.
4. Create read replicas and enable Database Proxy to split read and write requests. Read replicas can offload the read pressure from the primary instance, thus improving the database throughput. For details, see **Introducing Read/Write Splitting**.

5.9 CPU Resource Exhaustion Caused by Too Many Concurrent Slow Queries

Scenario

A large number of slow **select count(0)** operations are being concurrently executed on a DB instance. As a result, CPU resources are exhausted, causing system breakdown risks.

After your ran **Show processlist**, the command output showed that **select count(0)** operations are concurrently executed for multiple times.



```
nt,a.is_display as isDisplay,a.prise_num as priseNum,tag,a.createtime as crea |
| 4884 | root      | 192.168.1.11:63874  | vj_yiqi | Query          | 1550 | Sending data                                    | select a.id,title,conte
nt,a.is_display as isDisplay,a.prise_num as priseNum,tag,a.createtime as crea |
| 6797 | root      | 192.168.1.163:43789 | vj_yiqi | Query          | 1536 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 6800 | root      | 192.168.1.163:43791 | vj_yiqi | Query          | 1537 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 6905 | root      | 192.168.1.163:43809 | vj_yiqi | Query          | 1549 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7024 | root      | 192.168.1.163:43836 | vj_yiqi | Query          | 1519 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7040 | root      | 192.168.1.125:51791 | vj_yiqi | Query          |  112 | Sending data                                    | select a.id,title,conte
nt,a.is_display as isDisplay,a.prise_num as priseNum,tag,a.createtime as crea |
| 7049 | root      | 192.168.1.125:51792 | vj_yiqi | Query          |  846 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7050 | root      | 192.168.1.125:51793 | vj_yiqi | Query          |  727 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7053 | root      | 192.168.1.125:51794 | vj_yiqi | Query          |  818 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7054 | root      | 192.168.1.125:51795 | vj_yiqi | Query          | 1026 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7055 | root      | 192.168.1.125:51796 | vj_yiqi | Query          | 1436 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7056 | root      | 192.168.1.125:51797 | vj_yiqi | Query          | 1493 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7057 | root      | 192.168.1.125:51798 | vj_yiqi | Query          | 1401 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7058 | root      | 192.168.1.125:51799 | vj_yiqi | Query          |  441 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7061 | root      | 192.168.1.125:51800 | vj_yiqi | Query          |  583 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7071 | root      | 192.168.1.163:43843 | vj_yiqi | Query          | 1529 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7072 | root      | 192.168.1.163:43844 | vj_yiqi | Query          | 1549 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7073 | root      | 192.168.1.163:43845 | vj_yiqi | Query          | 1433 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7074 | root      | 192.168.1.163:43852 | vj_yiqi | Query          | 1474 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7075 | root      | 192.168.1.163:43853 | vj_yiqi | Query          | 1518 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7088 | root      | 192.168.1.11:64057  | vj_yiqi | Query          | 1080 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7089 | root      | 192.168.1.11:64058  | vj_yiqi | Query          |  646 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7090 | root      | 192.168.1.11:64059  | vj_yiqi | Query          |  322 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7091 | root      | 192.168.1.11:64060  | vj_yiqi | Query          | 1070 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7092 | root      | 192.168.1.11:64061  | vj_yiqi | Query          | 1146 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7093 | root      | 192.168.1.11:64062  | vj_yiqi | Query          | 1405 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7096 | root      | 192.168.1.11:64063  | vj_yiqi | Query          |  621 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7097 | root      | 192.168.1.11:64064  | vj_yiqi | Query          | 1300 | Sending data                                    | select count(0) from (s
elect a.id,title,content,a.is_display as isDisplay,a.prise_num as priseNum,ta |
| 7098 | root      | 192.168.1.11:64065  | vj_yiqi | Query          |  847 | Sending data                                    | select count(0) from (s
```

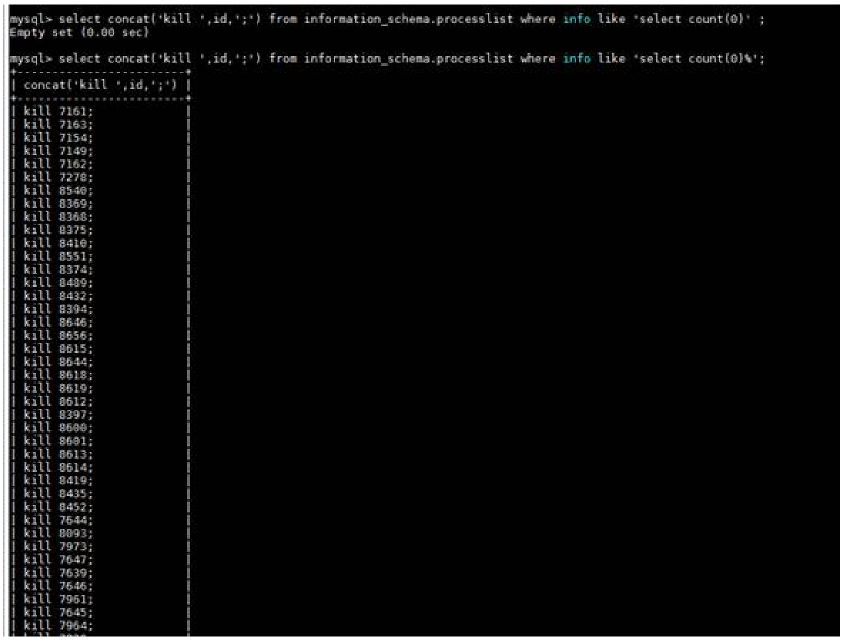
Possible Causes

Applications triggered a large number of concurrent and slow **select count(0)** operations, exhausting CPU resources.

Solution

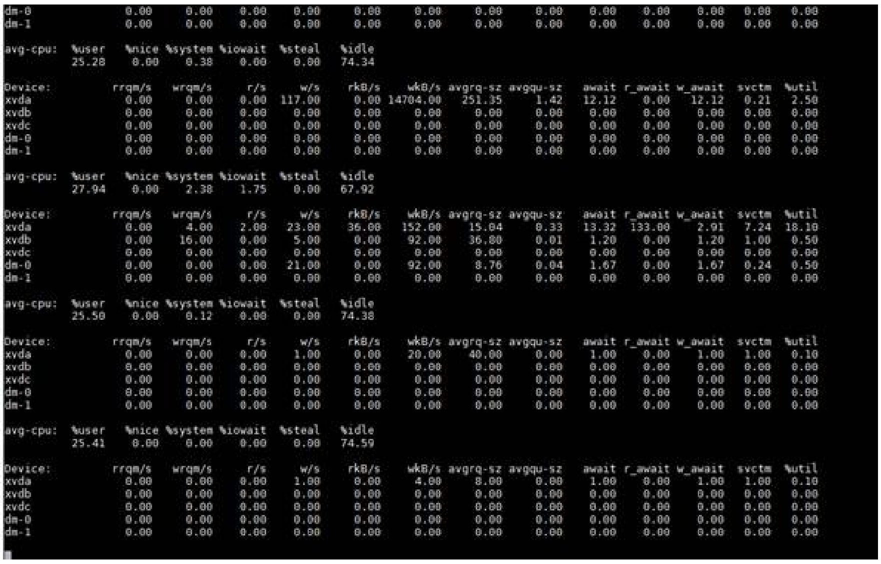
Step 1 Apply for permissions to intermittently kill the slow **select count(0)** operations in batches. Locate the source of triggering the slow operations, stop the source, and optimize SQL statements.

Kill the slow operations in batches.



```
mysql> select concat('kill ',id,';') from information_schema.processlist where info like 'select count(0)' ;
Empty set (0.00 sec)

mysql> select concat('kill ',id,';') from information_schema.processlist where info like 'select count(0)%';
+------------------------+
| concat('kill ',id,';') |
+------------------------+
| kill 7161;             |
| kill 7163;             |
| kill 7154;             |
| kill 7149;             |
| kill 7162;             |
| kill 7278;             |
| kill 8540;             |
| kill 8369;             |
| kill 8368;             |
| kill 8375;             |
| kill 8410;             |
| kill 8551;             |
| kill 8374;             |
| kill 8489;             |
| kill 8432;             |
| kill 8394;             |
| kill 8646;             |
| kill 8656;             |
| kill 8615;             |
| kill 8644;             |
| kill 8618;             |
| kill 8619;             |
| kill 8612;             |
| kill 8397;             |
| kill 8600;             |
| kill 8601;             |
| kill 8613;             |
| kill 8614;             |
| kill 8419;             |
| kill 8435;             |
| kill 8452;             |
| kill 7644;             |
| kill 8093;             |
| kill 7973;             |
| kill 7647;             |
| kill 7639;             |
| kill 7646;             |
| kill 7961;             |
| kill 7645;             |
| kill 7964;             |
```

Step 2 Increase CPU idle time.



```
dm-0              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-1              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          25.28    0.00    0.38    0.00    0.00   74.34

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
xvda              0.00     0.00    0.00  117.00     0.00 14704.00   251.35     1.42   12.12    0.00   12.12   0.21   2.50
xvdb              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
xvdc              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-0              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-1              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          27.94    0.00    2.38    1.75    0.00   67.92

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
xvda              0.00     4.00    2.00   23.00    36.00   152.00    15.04     0.33   13.32  133.00    2.91   7.24  18.10
xvdb              0.00    16.00    0.00    5.00     0.00    92.00    36.80     0.01    1.20    0.00    1.20   1.00   0.50
xvdc              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-0              0.00     0.00    0.00   21.00     0.00    92.00     8.76     0.04    1.67    0.00    1.67   0.24   0.50
dm-1              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          25.50    0.00    0.12    0.00    0.00   74.38

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
xvda              0.00     0.00    0.00    1.00     0.00    20.00    40.00     0.00    1.00    0.00    1.00   1.00   0.10
xvdb              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
xvdc              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-0              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-1              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          25.41    0.00    0.00    0.00    0.00   74.59

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await r_await w_await  svctm  %util
xvda              0.00     0.00    0.00    1.00     0.00     4.00     8.00     0.00    1.00    0.00    1.00   1.00   0.10
xvdb              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
xvdc              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-0              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
dm-1              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00    0.00    0.00   0.00   0.00
```

----End

6 Basic Issues

6.1 How Do I View the Used Storage of My TaurusDB Instance?

TaurusDB decouples compute and storage, so data is stored in the shared storage. You can view the used storage of your instance on the console, which is updated every 30 minutes.

NOTE

Storage is calculated differently for TaurusDB and open-source MySQL. To query the used storage of an open-source MySQL instance, run **select sum(data_length+index_length +data_free) from information_schema.tables;**

To query the used storage of a TaurusDB instance, log in to the TaurusDB console or run **show spaceusage;**

Viewing the Used Storage of a TaurusDB Instance on the Console

Step 1 Log in to the management console.

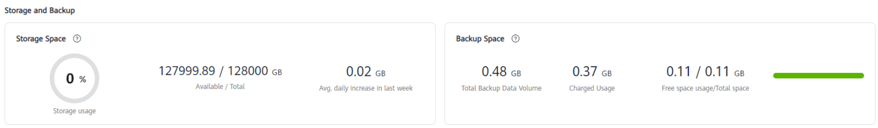
Step 2 Click in the upper left corner and select a region and project.

Step 3 Click ☰ in the upper left corner of the page and choose **Databases** > **TaurusDB**.

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

Step 5 In the **Storage and Backup** > **Storage Space** area, view the used storage of your instance.

Figure 6-1 Viewing the used storage of a DB instance



Storage and Backup

Storage Space

0 %
Storage usage

127999.89 / 128000 GB
Available / Total

0.02 GB
Avg. daily increase in last week

Backup Space

0.48 GB
Total Backup Data Volume

0.37 GB
Charged Usage

0.11 / 0.11 GB
Free space usage/Total space

- Storage space
  - The total and used storage of your yearly/monthly instance is displayed.
  - If this storage is used up, storage grows as data volume increases and you will be billed on a pay-per-use basis for any additional storage. To keep costs down, make sure you scale up storage in a timely manner so you can take advantage of yearly/monthly rates.
- Backup space:

  TaurusDB provides free backup storage equal to the amount of the used storage of your pay-per-use instances or the purchased storage of your yearly/ monthly instance.

----End

Viewing the Used Storage of a TaurusDB Instance Through the CLI

After connecting to a TaurusDB instance, run the following command to view its used storage:

show spaceusage;

The following table describes the calculation items of TaurusDB instance storage.

NOTE

information_schema cannot be updated in real time. Before a query, run the following command to update it:

set information_schema_stats_expiry = 0;

Table 6-1 Storage calculation items

Item	Command	Description
Table data	select sum(data_length +index_length+data_free) from information_schema.tables;	Size of table data. If the statistics data is not updated, the result may be inaccurate. Open-source MySQL uses this command to obtain the used storage of a DB instance.
Pre-allocated table space	select count(*) from information_schema.tables;	Pre-allocated space of tables. The statement is used to query the number of tables and the total pre-allocated space of tables is (Number of tables x 4 MB). The system pre-allocates 4 MB of storage to each table.

Item	Command	Description
Pre-allocated partition space	select count(*) from INFORMATION_SCHEMA.PARTITIONS where PARTITION_NAME is not null;	Pre-allocated space of partitions. The statement is used to query the number of partitions and the total pre-allocated space of partitions is (Number of partitions x 4 MB). The system pre-allocates 4 MB of storage to each partition.
Binlog	show binary logs;	Total size of all binlog files.
Redolog	show lsninfo;	flushed_to_disk_lsn-truncate_lsn
Undo space	select sum(INITIAL_SIZE) as undo_space from information_schema.files where file_type='UNDO LOG';	Undo space.

6.2 Renaming Databases and Tables

TaurusDB follows the same method as MySQL Community Edition for renaming databases and tables.

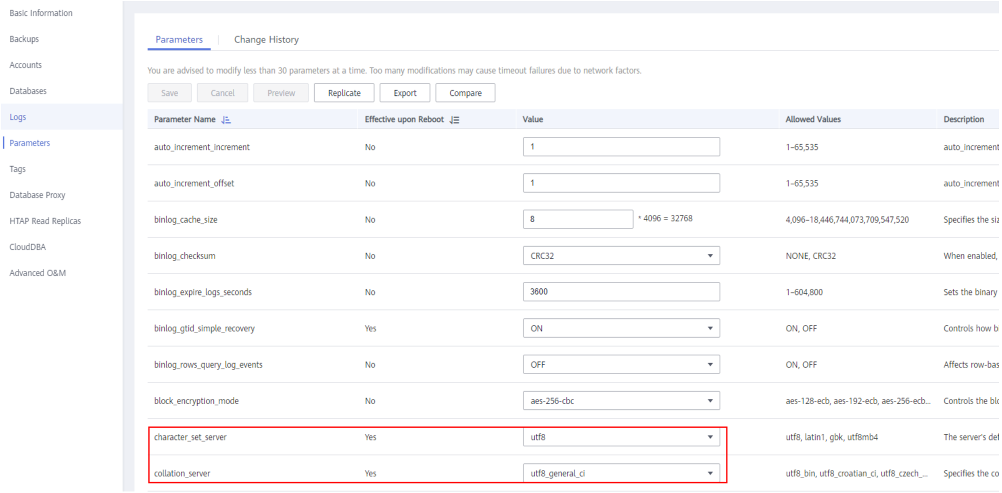
- Renaming table names: Run the **rename table** *a* **to** *b***;** statement. You can rename a table to move a table from one database to another. For example, **rename table da.ta to db.ta** moves the table **ta** from the database **da** to the database **db**.
- Renaming database names: Use the **rename table** command to move all tables from the original database to a new database, and then delete the original database. Example:

```
# Access the original database.
use ta;
# Show all table names in the original database.
Show tables;
# View the CREATE DATABASE statement in the original database.
Show create database ta;
# Use the CREATE DATABASE statement in the original database to create a new database. (Change only the database name.)
create database tb;
# Move all tables from the original database to the new database.
rename table da.ta to db.ta;
rename table da.tb to db.tb;
rename table da.tc to db.tc;
...
# Delete the original database.
Drop database ta;
```

6.3 Character Set and Collation Settings

Related Variables

By default, **character_set_server** is set to **utf8** and **collation_server** to **utf8_general_ci** for your DB instance. You can change the values on the console.



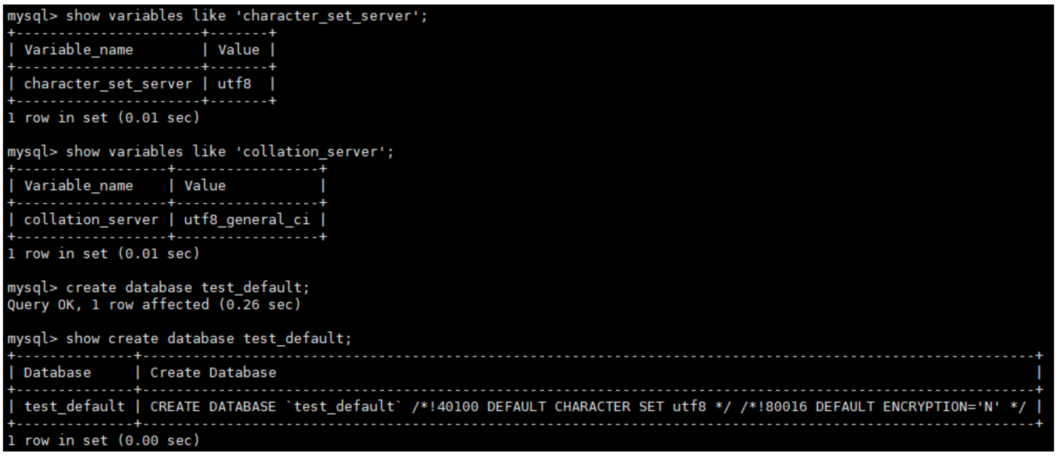
You are advised to modify less than 30 parameters at a time. Too many modifications may cause timeout failures due to network factors.

Parameter Name	Effective upon Reboot	Value	Allowed Values	Description
auto_increment_increment	No	1	1–65,535	auto_increment
auto_increment_offset	No	1	1–65,535	auto_increment
binlog_cache_size	No	8 * 4096 = 32768	4,096–18,446,744,073,709,547,520	Specifies the siz
binlog_checksum	No	CRC32	NONE, CRC32	When enabled,
binlog_expire_logs_seconds	No	3600	1–604,800	Sets the binary
binlog_gtid_simple_recovery	Yes	ON	ON, OFF	Controls how b
binlog_rows_query_log_events	No	OFF	ON, OFF	Affects row-bas
block_encryption_mode	No	aes-256-cbc	aes-128-ecb, aes-192-ecb, aes-256-ecb...	Controls the blc
character_set_server	Yes	utf8	utf8, latin1, gbk, utf8mb4	The server's def
collation_server	Yes	utf8_general_ci	utf8_bin, utf8_croatian_ci, utf8_czech_...	Specifies the co

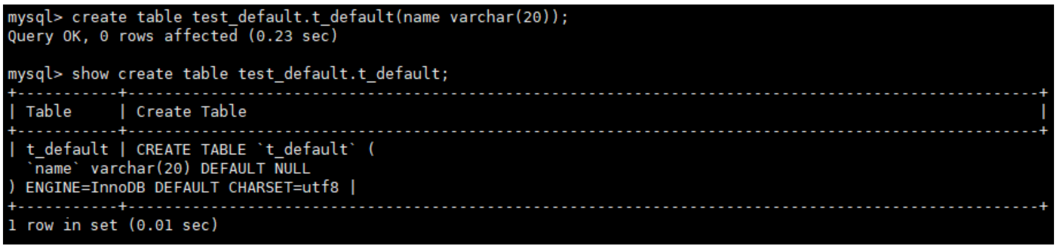
Configuring Character Sets and Collations for Databases, Tables, and Fields

- If the character set and collation are not explicitly specified for a database during database creation, the values of **character_set_server** and **collation_server** are used for the database. If the character set and collation are explicitly specified, the specified character set and collation are used for the database.
- If the character set and collation are not explicitly specified for a table during table creation, the character set and collation of the database hosting the table are used for the table. If the character set and collation are explicitly specified, the specified character set and collation are used for the table.
- If the character set and collation are not explicitly specified for a field during table creation, the character set and collation of the table hosting the field are used for the field. If the character set and collation are explicitly specified, the specified character set and collation are used for the field.

Example 1: Create a database and table without explicitly specifying the character set and collation.

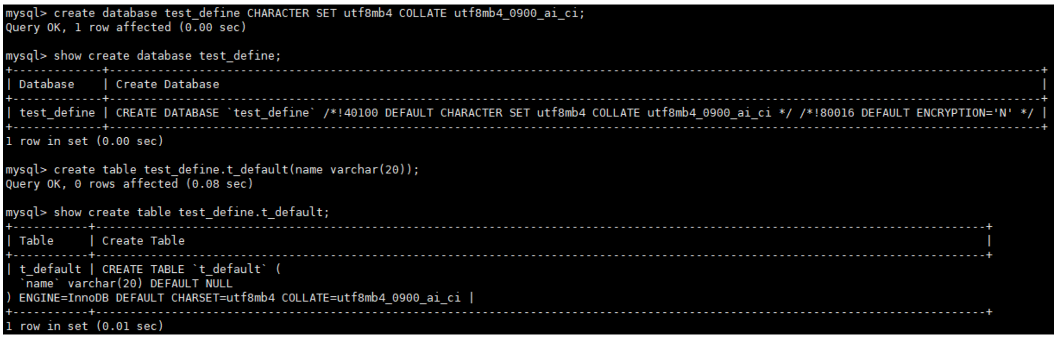


```
mysql> show variables like 'character_set_server';
+----------------------+-------+
| Variable_name        | Value |
+----------------------+-------+
| character_set_server | utf8  |
+----------------------+-------+
1 row in set (0.01 sec)

mysql> show variables like 'collation_server';
+------------------+-----------------+
| Variable_name    | Value           |
+------------------+-----------------+
| collation_server | utf8_general_ci |
+------------------+-----------------+
1 row in set (0.01 sec)

mysql> create database test_default;
Query OK, 1 row affected (0.26 sec)

mysql> show create database test_default;
+--------------+------------------------------------------------------------------------------------------------------------+
| Database     | Create Database                                                                                            |
+--------------+------------------------------------------------------------------------------------------------------------+
| test_default | CREATE DATABASE `test_default` /*!40100 DEFAULT CHARACTER SET utf8 */ /*!80016 DEFAULT ENCRYPTION='N' */ |
+--------------+------------------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```



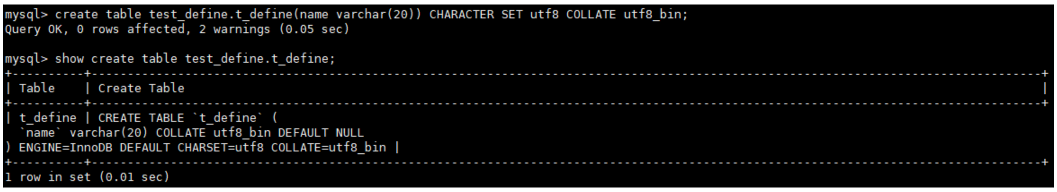
```
mysql> create table test_default.t_default(name varchar(20));
Query OK, 0 rows affected (0.23 sec)

mysql> show create table test_default.t_default;
+-----------+----------------------------------------------------------------------------------------------------+
| Table     | Create Table                                                                                       |
+-----------+----------------------------------------------------------------------------------------------------+
| t_default | CREATE TABLE `t_default` (
  `name` varchar(20) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 |
+-----------+----------------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)
```

Example 2: Create a database with the character set and collation explicitly specified.



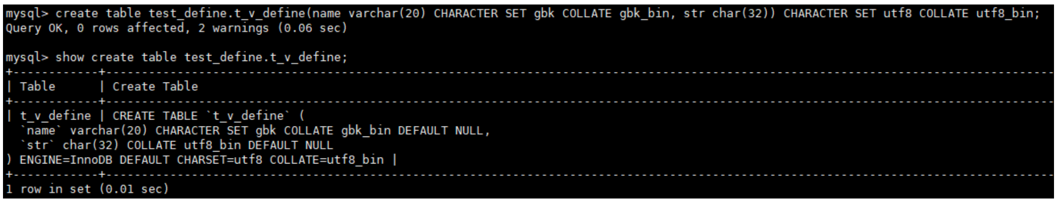
```
mysql> create database test_define CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci;
Query OK, 1 row affected (0.00 sec)

mysql> show create database test_define;
+-------------+---------------------------------------------------------------------------------------------------------------------------------------+
| Database    | Create Database                                                                                                                       |
+-------------+---------------------------------------------------------------------------------------------------------------------------------------+
| test_define | CREATE DATABASE `test_define` /*!40100 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci */ /*!80016 DEFAULT ENCRYPTION='N' */ |
+-------------+---------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)

mysql> create table test_define.t_default(name varchar(20));
Query OK, 0 rows affected (0.08 sec)

mysql> show create table test_define.t_default;
+-----------+---------------------------------------------------------------------------------------------------------------------+
| Table     | Create Table                                                                                                        |
+-----------+---------------------------------------------------------------------------------------------------------------------+
| t_default | CREATE TABLE `t_default` (
  `name` varchar(20) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+-----------+---------------------------------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)
```

Example 3: Create a table with the character set and collation explicitly specified.



```
mysql> create table test_define.t_define(name varchar(20)) CHARACTER SET utf8 COLLATE utf8_bin;
Query OK, 0 rows affected, 2 warnings (0.05 sec)

mysql> show create table test_define.t_define;
+----------+----------------------------------------------------------------------------------------------------------------------------+
| Table    | Create Table                                                                                                               |
+----------+----------------------------------------------------------------------------------------------------------------------------+
| t_define | CREATE TABLE `t_define` (
  `name` varchar(20) COLLATE utf8_bin DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin |
+----------+----------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)
```

Example 4: Create a table with the character set and collation explicitly specified for a field.



```
mysql> create table test_define.t_v_define(name varchar(20) CHARACTER SET gbk COLLATE gbk_bin, str char(32)) CHARACTER SET utf8 COLLATE utf8_bin;
Query OK, 0 rows affected, 2 warnings (0.06 sec)

mysql> show create table test_define.t_v_define;
+------------+----------------------------------------------------------------------------------------------------------------------------------
| Table      | Create Table
+------------+----------------------------------------------------------------------------------------------------------------------------------
| t_v_define | CREATE TABLE `t_v_define` (
  `name` varchar(20) CHARACTER SET gbk COLLATE gbk_bin DEFAULT NULL,
  `str` char(32) COLLATE utf8_bin DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin |
+------------+----------------------------------------------------------------------------------------------------------------------------------
1 row in set (0.01 sec)
```

6.4 Auto-Increment Field Value Jump

If the values of the auto-increment field are discontinuous, the possible causes are as follows:

- The increment is not 1.

```
mysql> show variables like 'auto_inc%';
+--------------------------+-------+
| Variable_name            | Value |
+--------------------------+-------+
| auto_increment_increment | 2     |
| auto_increment_offset    | 1     |
+--------------------------+-------+
mysql> select * from auto_test1;
+----+
| id |
+----+
|  2 |
|  4 |
|  6 |
|  8 |
+----+
```

- The value of **AUTO_INCREMENT** is changed.

```
mysql> select * from animals;
+----+-----------+
| id | name      |
+----+-----------+
|  1 | dog       |
|  2 | cat       |
|  3 | penguin   |
+----+-----------+
mysql> show create table animals;
+---------+-----------------------------------------------------+
| Table   | Create Table                                        |
+---------+-----------------------------------------------------+
| animals | CREATE TABLE `animals` (
`id` mediumint NOT NULL AUTO_INCREMENT,
`name` char(30) NOT NULL,
PRIMARY KEY (`id`) )
ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8 |
+---------+-----------------------------------------------------+
mysql> alter table animals AUTO_INCREMENT=100;
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0
mysql> show create table animals;
+---------+-----------------------------------------------------+
| Table   | Create Table                                   |
+---------+-----------------------------------------------------+
| animals | CREATE TABLE `animals` (
`id` mediumint NOT NULL AUTO_INCREMENT,
`name` char(30) NOT NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=100 DEFAULT CHARSET=utf8         |
+---------+-----------------------------------------------------+
mysql> INSERT INTO animals (id,name) VALUES(0,'rabbit');
Query OK, 1 row affected (0.00 sec)
mysql> select * from animals;
+-----+-----------+
| id  | name      |
+-----+-----------+
|   1 | dog       |
|   2 | cat       |
|   3 | penguin   |
| 100 | rabbit    |
+-----+-----------+
9 rows in set (0.00 sec)
```

- The value of the auto-increment field is specified when data is inserted.

```
mysql> select * from animals;
+----+-----------+
| id | name      |
+----+-----------+
|  1 | dog       |
|  2 | cat       |
|  3 | penguin   |
+----+-----------+
mysql> INSERT INTO animals (id,name) VALUES(100,'rabbit');
Query OK, 1 row affected (0.00 sec)
mysql> select * from animals;
+-----+-----------+
| id  | name      |
+-----+-----------+
|   1 | dog       |
|   2 | cat       |
|   3 | penguin   |
| 100 | rabbit    |
+-----+-----------+
9 rows in set (0.00 sec)
```

- If a transaction is not committed or is rolled back, the value of **AUTO_INCREMENT** increases but does not decrease after the transaction is rolled back. When data is inserted again, the value of the auto-increment field jumps.

```
mysql> show create table auto_test1;
+------------+----------------------------------------+
| Table      | Create Table                           |
+------------+----------------------------------------+
| auto_test1 | CREATE TABLE `auto_test1` (
`id` int NOT NULL AUTO_INCREMENT,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8 |
+------------+----------------------------------------+
1 row in set (0.00 sec)
mysql> select * from auto_test1;
+----+
| id |
+----+
|  1 |
|  2 |
|  3 |
+----+
mysql> begin;
Query OK, 0 rows affected (0.02 sec)
mysql> insert into auto_test1 values (0),(0),(0);
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0
mysql> select * from auto_test1;
+----+
| id |
+----+
|  1 |
|  2 |
|  3 |
|  4 |
|  5 |
|  6 |
+----+
6 rows in set (0.00 sec)
mysql> show create table auto_test1;
+------------+----------------------------------------+
| Table      | Create Table                           |
+------------+----------------------------------------+
| auto_test1 |
CREATE TABLE `auto_test1` (
`id` int NOT NULL AUTO_INCREMENT,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8 |
+------------+----------------------------------------+
1 row in set (0.00 sec)
mysql> rollback;
Query OK, 0 rows affected (0.05 sec)
mysql> select * from auto_test1;
+----+
| id |
+----+
|  1 |
|  2 |
|  3 |
+----+
3 rows in set (0.00 sec)
mysql> show create table auto_test1;
+------------+----------------------------------------+
| Table      | Create Table                           |
+------------+----------------------------------------+
| auto_test1 | CREATE TABLE `auto_test1` (
`id` int NOT NULL AUTO_INCREMENT,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8 |
+------------+----------------------------------------+
mysql> insert into auto_test1 values (0),(0),(0);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0
mysql> select * from auto_test1;
+----+
| id |
+----+
|  1 |
|  2 |
|  3 |
|  7 |
|  8 |
|  9 |
+----+
6 rows in set (0.00 sec)
mysql> show create table auto_test1;
+------------+-----------------------------------------+
| Table      | Create Table                            |
+------------+-----------------------------------------+
| auto_test1 | CREATE TABLE `auto_test1` (
`id` int NOT NULL AUTO_INCREMENT,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=10 DEFAULT CHARSET=utf8 |
+------------+----------------------------------------+
```

- After data is inserted, the value of **AUTO_INCREMENT** changes. But when the corresponding data row is deleted, the value of **AUTO_INCREMENT** does not decrease. When data is inserted again, the value of the auto-increment field jumps.

```
mysql> show create table auto_test1;
+------------+----------------------------------------+
| Table      | Create Table                           |
+------------+----------------------------------------+
| auto_test1 | CREATE TABLE `auto_test1` (
`id` int NOT NULL AUTO_INCREMENT,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8 |
+------------+----------------------------------------+
1 row in set (0.00 sec)
mysql> select * from auto_test1;
+----+
| id |
+----+
|  1 |
|  2 |
|  3 |
+----+
mysql> insert into auto_test1 values (0),(0),(0);
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0
mysql> select * from auto_test1;
+----+
| id |
+----+
|  1 |
|  2 |
|  3 |
|  4 |
|  5 |
|  6 |
+----+
6 rows in set (0.00 sec)
mysql> show create table auto_test1;
+------------+----------------------------------------+
| Table      | Create Table                           |
+------------+----------------------------------------+
| auto_test1 | CREATE TABLE `auto_test1` (
`id` int NOT NULL AUTO_INCREMENT,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8 |
+------------+----------------------------------------+
1 row in set (0.00 sec)
mysql> delete from auto_test1 where id>3;
mysql> select * from auto_test1;
+----+
| id |
+----+
|  1 |
|  2 |
|  3 |
+----+
3 rows in set (0.00 sec)
mysql> show create table auto_test1;
+------------+----------------------------------------+
| Table      | Create Table                           |
+------------+----------------------------------------+
| auto_test1 | CREATE TABLE `auto_test1` (
`id` int NOT NULL AUTO_INCREMENT,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8 |
+------------+----------------------------------------+
mysql> insert into auto_test1 values (0),(0),(0);
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0
mysql> select * from auto_test1;
+----+
| id |
+----+
|  1 |
|  2 |
|  3 |
|  7 |
|  8 |
|  9 |
+----+
6 rows in set (0.00 sec)
mysql> show create table auto_test1;
+------------+-----------------------------------------+
| Table      | Create Table                            |
+------------+-----------------------------------------+
| auto_test1 | CREATE TABLE `auto_test1` (
`id` int NOT NULL AUTO_INCREMENT,
PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=10 DEFAULT CHARSET=utf8 |
+------------+-----------------------------------------+
```

- If data insertion fails due to some reasons (for example, unique key conflict), the value of **AUTO_INCREMENT** may jump.

```
mysql> create table auto_test7(`id` int NOT NULL AUTO_INCREMENT, cred_id int UNIQUE, PRIMARY
KEY (`id`));
Query OK, 0 rows affected (0.64 sec)
mysql> insert into auto_test7 values(null, 1);
Query OK, 1 row affected (0.03 sec)
mysql> show create table auto_test7;
+------------+-------------------------------+
| Table      | Create Table                  |
+------------+-------------------------------+
| auto_test7 | CREATE TABLE `auto_test7` (  `id` int NOT NULL AUTO_INCREMENT,  `cred_id` int
DEFAULT NULL,  PRIMARY KEY (`id`),  UNIQUE KEY `cred_id` (`cred_id`)) ENGINE=InnoDB
AUTO_INCREMENT=2 DEFAULT CHARSET=utf8 |
+------------+---------------------------------------------------------------+
1 row in set (0.00 sec)
mysql> insert into auto_test7 values(null, 1);
ERROR 1062 (23000): Duplicate entry '1' for key 'auto_test7.cred_id'
mysql> show create table auto_test7;
+------------+---------------------------------------------------------------+
| Table      | Create Table                                                  |
+------------+---------------------------------------------------------------+
| auto_test7 | CREATE TABLE `auto_test7` (  `id` int NOT NULL AUTO_INCREMENT,  `cred_id` int
DEFAULT NULL,  PRIMARY KEY (`id`),  UNIQUE KEY `cred_id` (`cred_id`)) ENGINE=InnoDB
AUTO_INCREMENT=3 DEFAULT CHARSET=utf8 |
+------------+----------------------------------------------------------------+
```

- When data is inserted in batches (such as **insert...select** and **load file**), the auto-increment key is requested in batches. Two to the power of *n* sequence numbers are requested in each batch. If the sequence numbers are not used up, the sequence numbers will not be returned. As a result, the value of **AUTO_INCREMENT** may jump.

```
mysql> create table auto_test5_tmp(id tinyint not null AUTO_INCREMENT, name varchar(8), PRIMARY
KEY (`id`));
Query OK, 0 rows affected (0.08 sec)
mysql> select * from auto_test5;
+----+------+
| id | name |
+----+------+
|  1 | A    |
|  2 | B    |
|  3 | C    |
|  4 | X    |
|  5 | Y    |
|  6 | Z    |
|  8 | A    |
|  9 | B    |
| 10 | C    |
| 11 | X    |
| 12 | Y    |
| 13 | Z    |
+----+------+
12 rows in set (0.00 sec)
mysql> insert into auto_test5_tmp select 0,name from auto_test5;
Query OK, 12 rows affected (0.01 sec)
Records: 12  Duplicates: 0  Warnings: 0
mysql> select * from auto_test5_tmp;
+----+------+
| id | name |
+----+------+
|  1 | A    |
|  2 | B    |
|  3 | C    |
|  4 | X    |
|  5 | Y    |
|  6 | Z    |
|  7 | A    |
|  8 | B    |
|  9 | C    |
| 10 | X    |
| 11 | Y    |
| 12 | Z    |
+----+------+
12 rows in set (0.00 sec)
mysql> show create table auto_test5_tmp;
+----------------+------------------------------------------------------+
| Table          | Create Table                                         |
+----------------+------------------------------------------------------+
| auto_test5_tmp | CREATE TABLE `auto_test5_tmp` (  `id` tinyint NOT NULL AUTO_INCREMENT,
`name` varchar(8) DEFAULT NULL,  PRIMARY KEY (`id`)) ENGINE=InnoDB AUTO_INCREMENT=16
DEFAULT CHARSET=utf8 |
+----------------+------------------------------------------------------+
```

6.5 Starting Value and Increment of AUTO_INCREMENT

The starting value and increment of **AUTO_INCREMENT** are determined by the **auto_increment_offset** and **auto_increment_increment** parameters.

- **auto_increment_offset** determines the starting point for the AUTO_INCREMENT column value.
- **auto_increment_increment** controls the interval between successive column values.
- When the value of **auto_increment_offset** is greater than that of **auto_increment_increment**, the value of **auto_increment_offset** is ignored.
- When the value of **auto_increment_offset** is less than or equal to that of **auto_increment_increment**, the value of **AUTO_INCREMENT**:

  auto_increment_offset + *N* x **auto_increment_increment** (*N* indicates the number of inserted data records).

The default values of the two parameters in TaurusDB are **1**. To modify the parameters, perform the following steps:

Step 1 On the **Instances** page, click the instance name.

Step 2 In the navigation pane, choose **Parameters**. On the displayed page, modify parameters as required.

----End

For example:

1. If both **auto_increment_offset** and **auto_increment_increment** are set to **1**, the starting value is 1 and the increment is 1.

```
show variables like 'auto_inc%';
+--------------------------+-------+
| Variable_name            | Value |
+--------------------------+-------+
| auto_increment_increment | 1     |
| auto_increment_offset    | 1     |
+--------------------------+-------+
```

2. If **auto_increment_increment** is set to **2**, the increment is 2.

```
set session auto_increment_offset=2;
Query OK, 0 rows affected (0.02 sec)
show variables like 'auto_inc%';
+--------------------------+-------+
| Variable_name            | Value |
+--------------------------+-------+
| auto_increment_increment | 2     |
| auto_increment_offset    | 1     |
+--------------------------+-------+
```

3. If **auto_increment_offset** is set to **10** and **auto_increment_increment** is set to **2**, the starting value is 2 (because the value of **auto_increment_offset** is greater than that of **auto_increment_increment**) and the increment is 2.

```
set session auto_increment_offset=10;
set session auto_increment_increment=2;
show variables like 'auto_inc%';
+--------------------------+-------+
| Variable_name            | Value |
+--------------------------+-------+
| auto_increment_increment | 2     |
| auto_increment_offset    | 10    |
+--------------------------+-------+
create table auto_test2(id int NOT NULL AUTO_INCREMENT, PRIMARY KEY (`id`));
Query OK, 0 rows affected (0.08 sec)
show create table auto_test2;
CREATE TABLE `auto_test2` (   `id` int NOT NULL AUTO_INCREMENT,   PRIMARY KEY (`id`) )
ENGINE=InnoDB DEFAULT CHARSET=utf8
1 row in set (0.01 sec)
insert into auto_test2 values(0), (0), (0);
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0
select * from auto_test2;
+----+
| id |
+----+
|  2 |
|  4 |
|  6 |
+----+
3 rows in set (0.01 sec)
```

4. If **auto_increment_offset** is set to **5** and **auto_increment_increment** is set to **10**, the starting value is 5 and the increment is 10.

```
set session auto_increment_offset=5;
set session auto_increment_increment=10;
show variables like 'auto_inc%';
+--------------------------+-------+
| Variable_name            | Value |
+--------------------------+-------+
| auto_increment_increment | 10    |
| auto_increment_offset    | 5     |
+--------------------------+-------+
create table auto_test3(id int NOT NULL AUTO_INCREMENT, PRIMARY KEY (`id`));
insert into auto_test3 values(0), (0), (0);
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0
select * from auto_test3;
+----+
| id |
+----+
|  5 |
| 15 |
| 25 |
```

6.6 Changing the AUTO_INCREMENT Value of a Table

The methods are as follows:

1. If the value of **AUTO_INCREMENT** is greater than the maximum value of the auto-increment column in the table, **AUTO_INCREMENT** can be changed to a larger value within the value range.

```
show create table animals;
+---------+-------------------------------------------------------------------+
| Table   | Create Table                                                      |
+---------+-------------------------------------------------------------------+
| animals | CREATE TABLE `animals` (
`id` mediumint NOT NULL AUTO_INCREMENT,   `name` char(30) NOT NULL,
PRIMARY KEY (`id`) ) ENGINE=InnoDB AUTO_INCREMENT=101 DEFAULT CHARSET=utf8   |
+---------+-------------------------------------------------------------------+
1 row in set (0.00 sec)
mysql> select * from animals;
+-----+-----------+
| id  | name      |
+-----+-----------+
| -50 | -middle   |
|   1 | dog       |
|   2 | cat       |
|  50 | middle    |
| 100 | rabbit    |
+-----+-----------+
11 rows in set (0.00 sec)
alter table animals AUTO_INCREMENT=200;
Query OK, 0 rows affected (0.22 sec)
Records: 0  Duplicates: 0  Warnings: 0
 show create table animals;
+---------+-------------------------------------------------------+
| Table   | Create Table                                          |
+---------+-------------------------------------------------------+
| animals | CREATE TABLE `animals` (
`id` mediumint NOT NULL AUTO_INCREMENT,   `name` char(30) NOT NULL,
PRIMARY KEY (`id`) ) ENGINE=InnoDB AUTO_INCREMENT=200 DEFAULT CHARSET=utf8 |
+---------+-------------------------------------------------------+
```

2. If the new value of **AUTO_INCREMENT** is still greater than the maximum value of the auto-increment column in the table, the value change is successful. Otherwise, the value is changed to the maximum value of the auto-increment column plus 1 by default.

```
mysql> select * from animals;
+-----+-----------+
| id  | name      |
+-----+-----------+
| -50 | -middle   |
|   1 | dog       |
|   2 | cat       |
|  50 | middle    |
| 100 | rabbit    |
+-----+-----------+
mysql> show create table animals;
+---------+-----------------------------------------------------+
| Table   | Create Table                                        |
+---------+-----------------------------------------------------+
| animals | CREATE TABLE `animals` (
`id` mediumint NOT NULL AUTO_INCREMENT,   `name` char(30) NOT NULL,
PRIMARY KEY (`id`) ) ENGINE=InnoDB AUTO_INCREMENT=200 DEFAULT CHARSET=utf8 |
+---------+-----------------------------------------------------+
mysql> alter table animals AUTO_INCREMENT=150;
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
mysql> show create table animals;
+---------+-----------------------------------------------------+
| Table   | Create Table                                        |
+---------+-----------------------------------------------------+
| animals | CREATE TABLE `animals` (
`id` mediumint NOT NULL AUTO_INCREMENT,   `name` char(30) NOT NULL,
PRIMARY KEY (`id`) ) ENGINE=InnoDB AUTO_INCREMENT=150 DEFAULT CHARSET=utf8 |
+---------+-----------------------------------------------------+
mysql> alter table animals AUTO_INCREMENT=50;
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0
mysql> show create table animals;
+---------+-----------------------------------------------------+
| Table   | Create Table                                        |
+---------+-----------------------------------------------------+
| animals | CREATE TABLE `animals` (
`id` mediumint NOT NULL AUTO_INCREMENT,   `name` char(30) NOT NULL,
PRIMARY KEY (`id`) ) ENGINE=InnoDB AUTO_INCREMENT=101 DEFAULT CHARSET=utf8 |
+---------+-----------------------------------------------------+
mysql> delete from animals where id=100;
Query OK, 1 row affected (0.00 sec)
mysql> select * from animals;
+-----+-----------+
| id  | name      |
+-----+-----------+
| -50 | -middle   |
|   1 | dog       |
|   2 | cat       |
|  50 | middle    |
+-----+-----------+
10 rows in set (0.00 sec)
mysql> alter table animals AUTO_INCREMENT=50;
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0
mysql> show create table animals;
+---------+-----------------------------------------------------+
| Table   | Create Table                                        |
+---------+-----------------------------------------------------+
| animals | CREATE TABLE `animals` (
`id` mediumint NOT NULL AUTO_INCREMENT,   `name` char(30) NOT NULL,
PRIMARY KEY (`id`) ) ENGINE=InnoDB AUTO_INCREMENT=51 DEFAULT CHARSET=utf8 |
+---------+-----------------------------------------------------+
1 row in set (0.00 sec)
```

3. The value of **AUTO_INCREMENT** cannot be changed to a negative number.

```
alter table animals AUTO_INCREMENT=-1;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to
your MySQL server version for the right syntax to use near '-1' at line 1
```

6.7 Failed to Insert Data Because Values for the Auto-increment Primary Key Field Reach the Upper Limit

Scenario

The error message "ERROR 1062 (23000): Duplicate entry 'xxx' for key 'xxx'" was displayed when data was inserted into a table.

Possible Causes

The value for the auto-increment primary key field has reached the upper limit and cannot be increased. As a result, the auto-increment primary key value generated for the newly inserted data is the same as that of the previous data record in the table. Since the auto-increment primary key value cannot be duplicate, an error is reported.

Solution

1. If there are too many data changes and the actual data volume in the table is far less than the capacity of the auto-increment primary key, import all data

in the table to a new table, delete the original table, and change the name of the new table back to the original table. (There are multiple methods for importing and exporting data. The following is only an example.

a. Create the table **auto_test5_tmp**.

```
create table auto_test5_tmp(id tinyint not null AUTO_INCREMENT, name varchar(8), PRIMARY
KEY (`id`));
Query OK, 0 rows affected (0.07 sec)
```

b. Insert records into the table **auto_test5_tmp**.

```
insert into auto_test5_tmp select 0,name from auto_test5;
Query OK, 6 rows affected (0.01 sec)
Records: 6  Duplicates: 0  Warnings: 0
```

c. Query the data in **auto_test5_tmp**.

```
select * from auto_test5_tmp;
+----+------+
| id | name |
+----+------+
|  1 | A    |
|  2 | B    |
|  3 | C    |
|  4 | X    |
|  5 | Y    |
|  6 | Z    |
+----+------+
```

d. Deletes the original table **auto_test5**.

```
drop table auto_test5;
```

e. Rename the table **auto_test5_tmp** to **auto_test5**.

```
rename table auto_test5_tmp to auto_test5;
Query OK, 0 rows affected (0.12 sec)
```

2. If the value for the auto-increment primary key is too small, change the field type of the auto-increment primary key to store more data.

```
alter table auto_test6  modify column id int NOT NULL AUTO_INCREMENT;
Query OK, 6 rows affected (0.15 sec)
Records: 6  Duplicates: 0  Warnings: 0
```

6.8 Auto-increment Field Values

TaurusDB uses the following methods to assign values to an auto-increment field:

```
# Table structure
CREATE TABLE animals (
    id MEDIUMINT NOT NULL AUTO_INCREMENT,
    name CHAR(30) NOT NULL,
    PRIMARY KEY (id)
);
```

1. If no value is specified for the auto-increment field, TaurusDB automatically enters the value of **AUTO_INCREMENT** to the field.

a. Insert records into the table.

```
 INSERT INTO animals (name) VALUES ('dog'),('cat'),('penguin'),('lax'),('whale'),('ostrich');
```

b. Querying table data

```
 select * from animals;
+----+---------+
| id | name    |
+----+---------+
|  1 | dog     |
|  2 | cat     |
|  3 | penguin |
|  4 | lax     |
|  5 | whale   |
|  6 | ostrich |
+----+---------+
```

c. Query the table structure.

```
 show create table animals;
+---------+--------------------------------------------------+
| Table   | Create Table                          |
+---------+--------------------------------------------------+
| animals | CREATE TABLE `animals` (  `id` mediumint NOT NULL AUTO_INCREMENT,  `name` 
char(30) NOT NULL,  PRIMARY KEY (`id`)) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT 
CHARSET=utf8 |
+---------+--------------------------------------------------+
```

2. If **0** or **NULL** is specified for the auto-increment field, TaurusDB automatically enters the value of **AUTO_INCREMENT** to the field.

   a. Insert records into the table.

```
INSERT INTO animals (id,name) VALUES(0,'groundhog');
INSERT INTO animals (id,name) VALUES(NULL,'squirrel');
```

   b. Query data.

```
select * from animals;
+----+-----------+
| id | name      |
+----+-----------+
|  1 | dog       |
|  2 | cat       |
|  3 | penguin   |
|  4 | lax       |
|  5 | whale     |
|  6 | ostrich   |
|  7 | groundhog |
|  8 | squirrel  |
+----+-----------+
8 rows in set (0.00 sec)
```

   c. Query the table structure.

```
show create table animals;
+---------+------------------------------------------------------------------+
| Table   | Create Table                                                     |
+---------+------------------------------------------------------------------+
| animals | CREATE TABLE `animals` (  `id` mediumint NOT NULL AUTO_INCREMENT,  `name` 
char(30) NOT NULL,  PRIMARY KEY (`id`)) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT 
CHARSET=utf8 |
+---------+------------------------------------------------------------+
```

3. If the value *X* that is greater than the value of **AUTO_INCREMENT** is specified for the auto-increment field, TaurusDB inserts *X* to the field and changes **AUTO_INCREMENT** to *X* + 1.

   a. Insert records into the table.

```
INSERT INTO animals (id,name) VALUES(100,'rabbit');
```

   b. Query data.

```
select * from animals;
+-----+-----------+
| id  | name      |
+-----+-----------+
|   1 | dog       |
|   2 | cat       |
|   3 | penguin   |
|   4 | lax       |
|   5 | whale     |
|   6 | ostrich   |
|   7 | groundhog |
|   8 | squirrel  |
| 100 | rabbit    |
+-----+-----------+
9 rows in set (0.00 sec)
```

   c. Query the table structure.

```
show create table animals;
+---------+-----------------------------------------------------------------------+
| Table   | Create Table                                                                                |
+---------+-----------------------------------------------------------------------+
| animals | CREATE TABLE `animals` (  `id` mediumint NOT NULL AUTO_INCREMENT,  `name`
char(30) NOT NULL,  PRIMARY KEY (`id`)) ENGINE=InnoDB AUTO_INCREMENT=101 DEFAULT
CHARSET=utf8 |
+---------+-----------------------------------------------------------------------+
```

4. If a value less than the value of **AUTO_INCREMENT** is specified for the auto-increment field, TaurusDB enters the value to the field and **AUTO_INCREMENT** remains unchanged.

```
mysql> INSERT INTO animals (id,name) VALUES(50,'middle');
Query OK, 1 row affected (0.00 sec)
mysql> select * from animals;
+-----+-----------+
| id  | name      |
+-----+-----------+
|   1 | dog       |
|   2 | cat       |
|   3 | penguin   |
|   4 | lax       |
|   5 | whale     |
|   6 | ostrich   |
|   7 | groundhog |
|   8 | squirrel  |
|  50 | middle    |
| 100 | rabbit    |
+-----+-----------+
10 rows in set (0.00 sec)
mysql> show create table animals;
+---------+-------------------------------------------------------------------+
| Table   | Create Table                                                                                |
+---------+-------------------------------------------------------------------+
| animals | CREATE TABLE `animals` (  `id` mediumint NOT NULL AUTO_INCREMENT,  `name`
char(30) NOT NULL,  PRIMARY KEY (`id`)) ENGINE=InnoDB AUTO_INCREMENT=101 DEFAULT
CHARSET=utf8 |
+---------+-------------------------------------------------------------------+
```

5. If a negative value is specified for the auto-increment field, TaurusDB enters the value to the field and **AUTO_INCREMENT** remains unchanged.

   a. Insert records into the table.

   ```
   INSERT INTO animals (id,name) VALUES(-50,'-middle');
   ```

   b. Queries data.

   ```
   select * from animals;
   +-----+-----------+
   | id  | name      |
   +-----+-----------+
   | -50 | -middle   |
   |   1 | dog       |
   |   2 | cat       |
   |   3 | penguin   |
   |   4 | lax       |
   |   5 | whale     |
   |   6 | ostrich   |
   |   7 | groundhog |
   |   8 | squirrel  |
   |  50 | middle    |
   | 100 | rabbit    |
   +-----+-----------+
   11 rows in set (0.00 sec)
   ```

   c. Query the table structure.

   ```
   show create table animals;
   +---------+-------------------------------------------------------------------------------------+
   | Table   | Create Table
   +---------+-------------------------------------------------------------------------------------+
   | animals | CREATE TABLE `animals` (  `id` mediumint NOT NULL AUTO_INCREMENT,
   ```

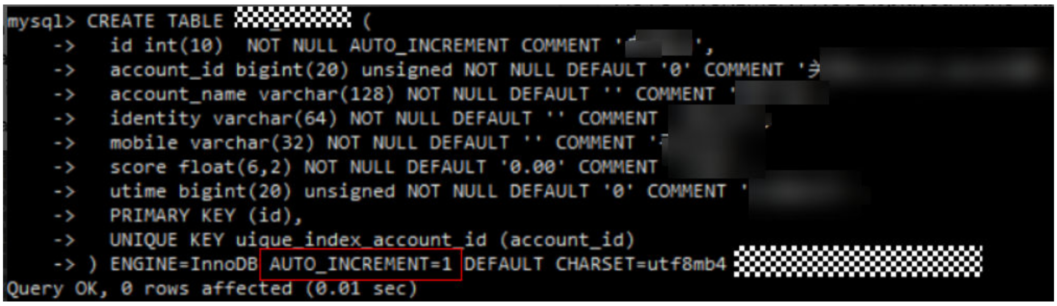
```
`name` char(30) NOT NULL,  PRIMARY KEY (`id`)) ENGINE=InnoDB AUTO_INCREMENT=101
DEFAULT CHARSET=utf8        |
+---------+--------------------------------------------------------------------------------------+
```

6.9 AUTO_INCREMENT Not Displayed in the Table Structure

Scenario

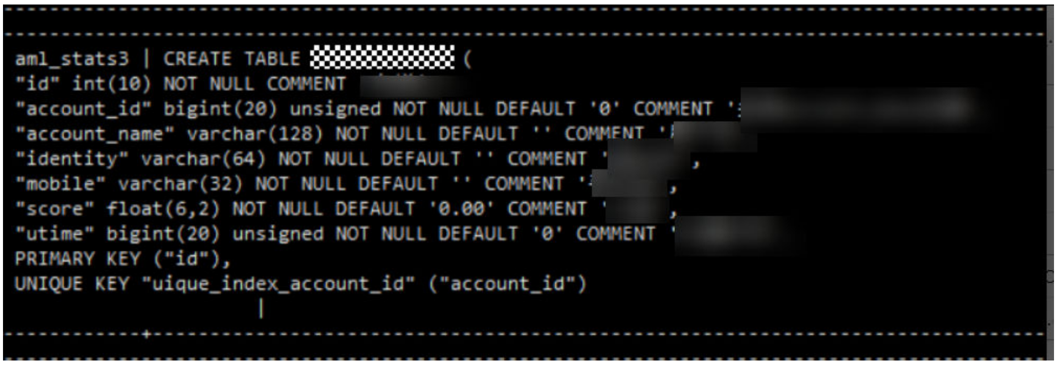
When a table was created, **AUTO_INCREMENT** was set to **1**. After **show create table** was executed, **AUTO_INCREMENT** was not displayed in the table structure.

A table was created:



```
mysql> CREATE TABLE [redacted] (
    ->   id int(10)  NOT NULL AUTO_INCREMENT COMMENT '[redacted]',
    ->   account_id bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '[redacted]
    ->   account_name varchar(128) NOT NULL DEFAULT '' COMMENT '[redacted]
    ->   identity varchar(64) NOT NULL DEFAULT '' COMMENT [redacted]
    ->   mobile varchar(32) NOT NULL DEFAULT '' COMMENT '[redacted]
    ->   score float(6,2) NOT NULL DEFAULT '0.00' COMMENT [redacted]
    ->   utime bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '[redacted]
    ->   PRIMARY KEY (id),
    ->   UNIQUE KEY uique_index_account_id (account_id)
    -> ) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4 [redacted]
Query OK, 0 rows affected (0.01 sec)
```

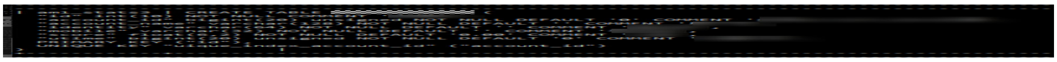
After **show create table** *xxx* was executed, **AUTO_INCREMENT** was not displayed in the table structure:



```
-----------------------------------------------------------------------------------------------

-----------------------------------------------------------------------------------------------
aml_stats3 | CREATE TABLE [redacted] (
"id" int(10) NOT NULL COMMENT [redacted]
"account_id" bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '[redacted]
"account_name" varchar(128) NOT NULL DEFAULT '' COMMENT '[redacted]
"identity" varchar(64) NOT NULL DEFAULT '' COMMENT '[redacted],
"mobile" varchar(32) NOT NULL DEFAULT '' COMMENT '[redacted],
"score" float(6,2) NOT NULL DEFAULT '0.00' COMMENT '[redacted],
"utime" bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '[redacted]
PRIMARY KEY ("id"),
UNIQUE KEY "uique_index_account_id" ("account_id")
                    |
-----------+-----------------------------------------------------------------------------------

-----------------------------------------------------------------------------------------------
```

Possible Causes

sql_mode was set to **NO_FIELD_OPTIONS**.



Valid values for **sql_mode** are as follows:

- **NO_FIELD_OPTIONS**: Do not print MySQL-specific column options in the output of SHOW CREATE TABLE.
- **NO_KEY_OPTIONS**: Do not print MySQL-specific index options in the output of SHOW CREATE TABLE.
- **NO_TABLE_OPTIONS**: Do not print MySQL-specific table options (such as ENGINE) in the output of SHOW CREATE TABLE.

Solution

Change the value of **sql_mode**.

6.10 Impact of Creating an Empty Username

The username **''** is allowed in TaurusDB instances, but using such an empty username has negative impacts on instances.

When you perform operations on a TaurusDB instance using an empty username, any username can be matched. This brings security and functional impacts on your instance. You are advised not to use empty usernames in actual scenarios.

- Security impact
  - Your instance can be connected using any username if there is an empty username.
  - Your database can be logged in using any username and the password of the empty username and the login user will obtain all permissions of the empty username. For example:

```
#If there is no empty username created and the invalid username abcd is used to connect to
the instance, the connection fails.
mysql> select user,host from mysql.user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| root             | %         |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
+------------------+-----------+
mysql -uabcd -h127.0.0.1 -P3306 -pTest_1234
mysql: [Warning] Using a password on the command line interface can be insecure.
ERROR 1045 (28000): Access denied for user 'abcd'@'localhost' (using password: YES)

#If an empty username has been created and the invalid username abcd and the password of
the empty username are used to connect to the instance, the connection is successful.
mysql> create user ''@'localhost' IDENTIFIED BY 'Test_1234';
mysql> select user,host from mysql.user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| root             | %         |
|                  | localhost |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
+------------------+-----------+
mysql -uabcd -h127.0.0.1 -P3306 -pTest_1234
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 37Server version: 8.0.22-debug Source distribution
Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates.
Other names may be trademarks of their respective owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql>
```

  - If the empty user does not have a password, you can use any username to log in to the instance without a password and obtain all permissions of the empty user. For example:

```
#If there is an empty username that does not have a password, the database can be logged in
using any username without a password.
mysql> create user ''@'localhost';
```

```
Query OK, 0 rows affected (8.87 sec)
mysql> select user,host from mysql.user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| root             | %         |
|                  | localhost |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
+------------------+-----------+
mysql -uabcd -h127.0.0.1 -P3306
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 39Server version: 8.0.22-debug Source distribution
Copyright (c) 2000, 2020, Oracle and/or its affiliates.
All rights reserved. Oracle is a registered trademark of Oracle Corporation and/or its affiliates.
Other names may be trademarks of their respective owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql>
#-----------------
mysql -usdhsjkdshk -h127.0.0.1 -P3306
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 40Server version: 8.0.22-debug Source distribution
Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates.
Other names may be trademarks of their respective owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql>
```

- Functional impact

  If there is an empty username, the database cannot be logged in using a correct username due to a name matching error.

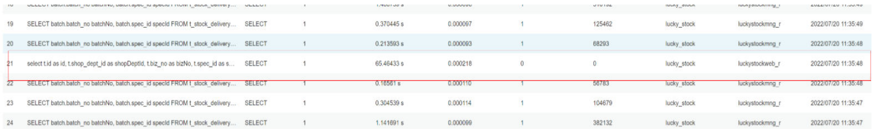
  Example: If the host of an empty user overlaps that of the **root** user, the **root** user cannot log in to the database using its password or it can log in to the database using the password of the empty username but cannot obtain the **root** user permissions.

```
mysql> create user ''@'localhost';
Query OK, 0 rows affected (8.87 sec)
mysql> select user,host from mysql.user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| root             | %         |
|                  | localhost |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
+------------------+-----------+
#The database cannot be logged in using the password of the root user.
mysql -uroot -h127.0.0.1 -P3306 -pTest_root
mysql: [Warning] Using a password on the command line interface can be insecure.
ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: YES)
#The user who logs in to the database using the password of the empty user (password-free) is
actually an empty user so the user does not have the root permissions.
mysql -uroot -h127.0.0.1 -P3306
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 45Server version: 8.0.22-debug Source distribution
Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates.
Other names may be trademarks of their respective owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql> select user,host from mysql.user;
ERROR 1142 (42000): SELECT command denied to user ''@'localhost' for table 'user'
```

6.11 No Scanned Rows Recorded in Slow Query Logs

Scenario

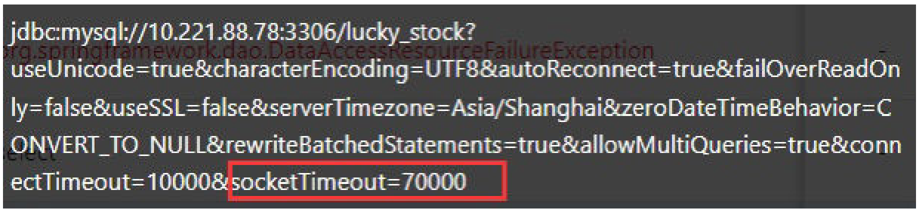
In slow query logs, an SQL statement was executed for 65 seconds, but the number of scanned rows was 0.



[illegible]	[illegible]	[illegible]	1	[illegible]	[illegible]	1	[illegible]	lucky_stock	luckystockmng_r	[illegible]
19	SELECT batch.batch_no batchNo, batch.spec_id specId FROM t_stock_delivery...	SELECT	1	0.370445 s	0.000097	1	125462	lucky_stock	luckystockmng_r	2022/07/20 11:35:49
20	SELECT batch.batch_no batchNo, batch.spec_id specId FROM t_stock_delivery...	SELECT	1	0.213593 s	0.000093	1	68293	lucky_stock	luckystockmng_r	2022/07/20 11:35:48
21	select t.id as id, t.shop_dept_id as shopDeptId, t.biz_no as bizNo, t.spec_id as s...	SELECT	1	65.46433 s	0.000218	0	0	lucky_stock	luckystockweb_r	2022/07/20 11:35:48
22	SELECT batch.batch_no batchNo, batch.spec_id specId FROM t_stock_delivery...	SELECT	1	0.16561 s	0.000110	1	56783	lucky_stock	luckystockmng_r	2022/07/20 11:35:48
23	SELECT batch.batch_no batchNo, batch.spec_id specId FROM t_stock_delivery...	SELECT	1	0.304539 s	0.000114	1	104679	lucky_stock	luckystockmng_r	2022/07/20 11:35:47
24	SELECT batch.batch_no batchNo, batch.spec_id specId FROM t_stock_delivery...	SELECT	1	1.141691 s	0.000099	1	382132	lucky_stock	luckystockmng_r	2022/07/20 11:35:47

Possible Causes

If an SQL statement is interrupted but its execution time exceeds the slow log threshold, the statement will be recorded in slow query logs and the number of scanned rows is 0. Timeout thresholds have been configured for the JDBC connection from the client.



```
jdbc:mysql://10.221.88.78:3306/lucky_stock?
useUnicode=true&characterEncoding=UTF8&autoReconnect=true&failOverReadOn
ly=false&useSSL=false&serverTimezone=Asia/Shanghai&zeroDateTimeBehavior=C
ONVERT_TO_NULL&rewriteBatchedStatements=true&allowMultiQueries=true&conn
ectTimeout=10000&socketTimeout=70000
```

Solution

Optimize the SQL statement or set **socketTimeout** to an appropriate value.

6.12 "handle_sync_msg_from_slave my_net_read error:-1" Displayed on the Error Logs Tab Page

Scenario

The error message "handle_sync_msg_from_slave my_net_read error:-1" was displayed on the **Error Logs** tab.

Possible Causes

During primary/standby synchronization, network data packet errors may occasionally occur due to network jitter. In this case, the system automatically retries.

Solution

No action is required.

6.13 ERROR 1290 (HY000): The MySQL server is running with the --sql-replica-on option so it cannot execute this statement

Scenario

When an SQL statement was executed, the following error information was displayed:

```
mysql> create database test;
ERROR 1290 (HY000): The MySQL server is running with the --sql-replica-on option so it cannot execute
this statement
```

Possible Causes

A read replica received requests to add, delete, and modify data, but the **sql_replica_on** parameter was set to on **to** prohibit these operations.

Solution

Check whether the client connection IP address for write operations is correct. You are advised to connect to the floating IP address of the DB instance or the read/write IP address of the proxy instance.