TDSQL-C for MySQL

Kernel Features

Product Documentation

Contents

Kernel Features
- Kernel Overview
- Kernel Version Release Notes
- Functionality Features
  - Instant DDL Overview

Kernel Features
Kernel Overview

Last updated：2022-06-13 11:32:53

The engine kernel of TDSQL-C for MySQL is fully compatible with native MySQL. You can migrate MySQL data to TDSQL-C for MySQL without modifying any application code or configuration.

The engine kernel of TDSQL-C for MySQL provides various features similar to those in MySQL Enterprise Edition, including enterprise-level transparent data encryption, audit, thread pool, encryption function, and backup and restoration.

The engine kernel of TDSQL-C for MySQL not only deeply optimizes the InnoDB storage engine and query performance, but also improves the ease of use and maintainability of databases. While providing all the benefits of MySQL, it offers more enterprise-grade advanced features such as disaster recovery, restoration, monitoring, performance optimization, read/write separation, transparent data encryption, and database audit.

For more information on the engine kernel of TDSQL-C for MySQL, see:

- For updates of the TDSQL-C for MySQL kernel engine version, see Kernel Version Release Notes.
- The kernel minor version of TDSQL-C for MySQL can be upgraded automatically or manually. For more information, see Upgrading Kernel Minor Version.

Release dates of different TDSQL-C for MySQL kernel versions:

TDSQL-C for MySQL 8.0

Kernel Version	Release Date
3.1.2	February 2022
3.1.1	November 2021
3.0.1	August 2021

TDSQL-C for MySQL 5.7

Kernel Version	Release Date
2.0.16	January 2022
2.0.15	October 2021
2.0.14	July 2021

Kernel Version	Release Date
2.0.13	March 2021
2.0.12	November 2020
2.0.11	June 2020

Kernel Version Release Notes

Last updated：2022-09-20 00:03:26

This document describes the TDSQL-C for MySQL kernel version updates. For information on how to upgrade the kernel, see Upgrading Kernel Minor Version.

TDSQL-C for MySQL 8.0

3.1.3

Feature updates

- Supported adding the binlog with the specified filename to an index file.
- Added a backup lock in the syntax of LOCK TABLES FOR BACKUP, UNLOCK TABLES.
- Added a binlog lock in the syntax of LOCK BINLOG FOR BACKUP, UNLOCK BINLOG.

Fixes

- Fixed the bug where dynamic metadata persistence caused instance table corruptions or visibility errors.
- Merged the official bugfix #32897503 to solve the issue where the execution path of some query statements was incorrect under the `prepare` statement.
- Merged the official bugfix to solve the crash when `set resource group` failed.
- Fixed the bug where the previous `gtid` was empty after HA switch.
- Fixed the bug where an auto-increment column could be set to a value smaller than the inserted maximum value.
- Fixed the issue where explicit transactions in read-only instances would block the replay thread from replaying DDL logs.
- Fixed the issue where tables with "-" in the name might crash when replicated in a read-only instance after DDL.
- Fixed the crash caused by the `undo` request to the DDL log system table when a read-only instance experienced DDL recovery upon startup.

3.1.2

Feature updates

- Supported MySQL 8.0 for read-only nodes and source-replica physical replication.
- Supported table space expansion and up to above 1 PB of capacity per instance.
- Supported limiting the number of preloaded rows, which achieved a 1%-5% performance increase during point query testing.

- Supported extended ANALYZE syntax (UPDATE HISTOGRAM c USING DATA 'json') and direct writes to histograms.

Performance optimizations

- Replaced index dive with histogram to reduce evaluation errors and I/O overheads (this capability is not enabled by default).

Fixes

- Fixed the issue where updates related to large object pages were not written to the log when a full-text index containing large object columns was created.
- Fixed the issue with inconsistent formats of `undo page` and different definitions of `TRX_UNDO_HISTORY_NODE` in the computing and storage layers.
- Fixed the issue where statistics information might be zero during online-DDL.
- Fixed the issue where columns generated from replica instances were not updated.
- Fixed the issue where the instance hung when binlog was compressed.
- Fixed the issue of missing GTID in the previous_gtids event of the newly generated binlog file.
- Fixed possible deadlocks when system variables were modified.
- Fixed the issue where the information of the SQL thread of the replica instance in SHOW PROCESSLIST was incorrectly displayed.
- Implemented the bug fix related to hash join provided in MySQL 8.0.23.
- Implemented the bug fix related to writeset provided in MySQL.
- Implemented the bug fix related to the query optimizer provided in MySQL 8.0.24.
- Fixed the concurrency bugs of optimizing flush list and releasing pages in FAST DDL.
- Optimized the memory usage during data dictionary update in instances with a large number of tables.
- Fixed the crash caused by new primary key creation after INSTANT ADD COLUMN.
- Fixed the OOM caused by memory growth in full-text index query.
- Fixed the issue where -1 was included in the TIME field in the result set returned by SHOW PROCESSLIST.
- Fixed the issue where tables might fail to be opened due to histogram compatibility.
- Fixed the floating point accumulation error when Singleton histograms were constructed.
- Fixed the replication interruption caused by using many Chinese characters in the table name of a row format log.

3.1.1

Feature updates

- Supported the official updates of MySQL 8.0.19, 8.0.20, 8.0.21, and 8.0.22.
- Supported dynamic setting of thread pooling mode or connection pooling mode by using the `thread_handling` parameter.

- Supported source-replica buffer pool sync: After a high-availability (HA) source-replica switch occurs, it usually takes a long time to warm up the replica, that is, to load hotspot data into its buffer pool. To accelerate the replica's warmup, TXSQL now supports the buffer pool sync between the source and the replica.
- Supported sort merge join.
- Supported async commit: With the thread pool enabled and binlog disabled, async commit can be enabled by setting the parameter `innodb_log_sync_method` to `async`.
- Supported fast DDL operations.
- Supported querying the value of the `character_set_client_handshake` parameter.
- Supported database audit.

Performance optimizations

- Optimized the mechanism of scanning and flushing the dirty pages tracked in the flush list, so as to solve the performance fluctuation issue while creating indexes and thus improve the system stability.
- Optimized the `BINLOG LOCK_done` conflict to improve write performance.
- Optimized the `trx_sys mutex` conflict by using lock free hash and improve performance.
- Optimized redo log flushing.
- Optimized the buffer pool initialization time.
- Optimized the clearing of adaptive hash indexes (AHI) during the `drop table` operations on big tables.

Fixes

- Fixed the deadlocks caused by the modification of the `offline_mode` and `cdb_working_mode` parameters.
- Fixed the concurrency issue while persistently storing `max_trx_id` of global object `trx_sys`.
- Fixed performance fluctuation when cleaning InnoDB temporary tables.
- Fixed the read-only performance decrease when the instance has many cores.
- Fixed the error (error code: 1032) caused by hash scans.
- Fixed concurrency security issues caused by hotspot update.

3.0.1

Feature updates

- Supported three methods of querying `cynos_version`: `select CYNOS_VERSION()`, `select @@cynos_version`, and `show variables like 'cynos_version'`.
- Added a space limit parameter. If the total space usage exceeds the limit, an error will be reported to prompt you to release the space or upgrade the specification.
- Added the `innodb_ncdb_log_priority` read-only parameter, which indicates the priority of the source instance's backend log thread.

- Added the `innodb_ncdb_apply_priority` read-only parameter, which indicates the priority of the read-only instance's log replay thread.
- Added the `innodb_ncdb_fast_shutdown` dynamic parameter, which controls whether to quickly shut down processes. After it is enabled, when a process exits, no destruction operations on the global structure will be performed, which reduces the shutdown time. It is disabled by default.
- Added the `innodb_max_temp_data_file_size` read-only parameter. Its default value is 128 MB. If its value is greater than 0, it indicates the maximum size of the temp tablespace in the local storage.

TDSQL-C for MySQL 5.7

2.0.17

Feature updates

- Supported adding the binlog with the specified filename to an index file.

Fixes

- Merged the official bugfix #25865525 to solve the issue where `LOAD DATA INFILE` failed to read escape characters plus UTF8 characters.

2.0.16

Feature updates

- Optimized `undo space truncate` to improve the speed of `undo truncate` on large-spec instances.
- Optimized the performance of large-scale queries on read-only instances.

Fixes

- Fixed the issue where `backup lock` couldn't be locked due to the `lock table` statement.
- Fixed the issue where `table share` went wrong after thousands of columns were added through `instant add` .
- Fixed the replay error when the content of binlog contained escaped keywords.
- Fixed the issue where externally prepared XA transactions were not explicitly rolled back and thus blocked normal shutdown.
- Fixed the issue where the warning "tablespace -1 not found" was reported during read-only instance startup.

2.0.15

Feature updates

- Supported the extended table space: When a single table space exceeds the `innodb_ncdb_extend_space_threshold` configuration, a new table will be created in the extended table space.
- Added new JSON functions: JSON_MERGE_PRESERVE, JSON_MERGE_PATCH, JSON_PRETTY, JSON_STORAGE_SIZE, JSON_ARRAYAGG, JSON_OBJECTAGG.
- Optimized the table lock recovery process at system startup to shorten the startup time.

Fixes

- Fixed the bugs for the `group by` performance issue in text columns and multiple issues related to virtual columns.
- Fixed the possible crash when large transaction rollback and shutdown operations were performed concurrently after instance startup.
- Fixed the issue where repeatedly failed IO retries of related pages caused instance exit after `undo space truncate` failed.
- Fixed the crash when statistics update accessed the snapshot cache for disabled read-only instances.
- Fixed the issue where the truncation log might be behind the `truncate` operation in `undo space truncate` .
- Fixed the bug where an error in ICP check for partitioned table scan resulted in slow query.
- Fixed the crash when the previous scan in a read-only instance encountered the partial replay of a split index log.

2.0.14

Feature updates

- Supported INSTANT MODIFY COLUMN. For more information, see Instant DDL Overview.

Fixes

- Fixed the issue where the used space was not reclaimed when a temp table in a read-only instance was dropped.
- Fixed the issue where the process exited when a read-only instance read the old page version due to changes in storage routes.
- Fixed the issue of possible crash when `information_schema.metadata_locks` was queried.
- Fixed the concurrency error of forward scan and B-tree SMO in read-only instance.
- Fixed the issue where when multiple tables had complex foreign key dependencies and the foreign key attribute was `ON DELETE CASCADE` , the corresponding record in a child table might be deleted twice when a record was delete in its parent table with DELETE.
- Fixed the issue where the process exited due to operations such as CREATE USER in a read-only instance.
- Fixed the issue where SHOW VOLUME STATUS in a read-only instance might trigger an assertion failure.

- Fixed the issue where a read-only instance had abnormal query results and crashed when DDL operations were performed in partitioned tables frequently.
- Fixed the issue where the process crashed during historical version construction when an read-only instance scanned a purged undo log.

2.0.13

Feature updates

- Supported INSTANT ADD COLUMN. For more information, see Instant DDL Overview.
- Optimized the audit performance under high load and added the `lock_usleep_time` dynamic parameter.

Fixes

- Fixed the issue where `dict_operation_lock` might cause deadlock during foreign key check.
- Fixed the issue where the process might exit when the ACL change log was replayed during read-only instance startup.
- Fixed the issue where data in source and replica instances was inconsistent after INSTANT ADD COLUMN and TRUNCATE TABLE.
- Fixed the log replay error occurring when data was inserted again after INSTANT ADD COLUMN and TRUNCATE TABLE.
- Fixed the issue where the process exited when a primary key containing a column added by INSTANT ADD COLUMN was created.
- Fixed the issue where the process exited during table structure query in a read-only instance when INSTANT COLUMN was used to create or rebuild a partition.

2.0.12

Feature updates

- Supported database audit. For more information on how to use it, see Enabling TDSQL-C Audit.
- Supported purging page read-ahead to accelerate space reclaim.
- Supported real-time update of the size information of tables and indexes in the system view.
- Added a thread to accelerate recycle LSN and storage GC.

Fixes

- Fixed official bugs in full-text index, including BUG#24938374, BUG#21625016, BUG#27082268, BUG#27155294, BUG#27326796, BUG#27304661, BUG#25289359, BUG#29717909, and BUG#30787535.
- Fixed official bugs where concurrent update might cause system crashes, including BUG#30950714, BUG#31205266, and BUG#25669686.

- Fixed the official bug #28104394 where uncommitted INSERT operations would affect the range scan created by an index and made it return an incorrect number of rows.
- Fixed the official bug #30488700 where an incorrect query execution plan of a derived table could result in a poor performance.
- Fixed the issue where the process exited when a read-only instance replayed statistics logs after the source (read-write) instance executed a DDL statement.
- Fixed the issue where RENAME TABLE was performed on a database that did not exist.
- Fixed the issue where a read-only instance might exit when OPTIMIZE TABLE was used for the source instance.
- Fixed the issue where transactions were inconsistent after a table with a full-text index was updated in a read-only instance.
- Fixed the deadlock occurring during SET OFFLINE MODE and SHOW VARIABLES operations.

2.0.11

Feature updates

- Optimized the binlog file index to accelerate binlog file scan.
- Optimized the shutdown process to make it faster.
- Optimized the instance memory to reduce the memory usage by structures such as buffer, ZIP, and hash.
- Optimized the DDL lock recovery process during read-only instance startup to accelerate replay.
- Optimized system table loading in read-only instance to accelerate startup.
- Optimized worker thread assignment during PURGE COORDINATOR to accelerate purge.

Fixes

- Fixed the issue where the process crashed during OPEN TABLE due to incorrect index structure mapping.
- Fixed the issue where read-only instance replication was abnormal during XA transaction rollback.
- Fixed the issue where startup crashed when binlog replication was started in a read-only instance.
- Fixed the memory leak caused by FLUSH LOGS.
- Fixed the shutdown failure of `mysqladmin shutdown` .
- Fixed the issue where a read-only instance failed to replay logs when DROP COLUMN was performed on the source instance.
- Fixed the issue where data could still be input after space restriction was triggered when a BLOB was inserted.
- Fixed the issue of read-only instance replication availability that might be caused by DDL statements in big tables or slow log storage in the source instance.
- Fixed the issue where storage wasted small tables after `innodb_max_temp_data_file_size` was set for a temp table.

Functionality Features
Instant DDL Overview

Last updated：2022-12-01 11:21:44

Use Cases

This feature can use DDL operations to alter ultra big tables in online businesses within seconds.

Feature Overview

The instant DDL feature can quickly modify columns in big tables while avoiding data replication. This feature does not replicate the data or consume disk capacity or I/O, and can implement changes within seconds during peak hours.

Supported Versions

- MySQL 5.7 with kernel minor version 2.0.13 or later.
- MySQL 8.0 with kernel minor version 3.1.1 or later.

Notes

Instant DDL supports the following operations:

- ADD COLUMN
- MODIFY COLUMN

INSTANT ADD COLUMN operation description

1. INSTANT ADD COLUMN syntax
   Add the `algorithm=instant` clause to `ALTER TABLE` to add a column as follows:

```
ALTER TABLE t1 ADD COLUMN c INT, ADD COLUMN d INT DEFAULT 1000, ALGORITHM=INSTA
NT;
```

2. The `innodb_alter_table_default_algorithm` parameter is added, which can be set to `inplace` or `instant`.
   This parameter is `inplace` by default and can be configured to adjust the default algorithm of `ALTER TABLE` as follows:

```
SET @@global.innodb_alter_table_default_algorithm=instant;
```

If no algorithm is specified, the default algorithm configured by this parameter will be used for `ALTER TABLE` operations.

Restrictions on INSTANT ADD COLUMN

- A statement can contain only column addition operations.
- A new column will be added to the end, and column order cannot be changed.
- INSTANT ADD COLUMN is not supported in tables with the row format being COMPRESSED.
- INSTANT ADD COLUMN is not supported in tables with a full-text index.
- INSTANT ADD COLUMN is not supported for temp tables.

INSTANT MODIFY COLUMN operation description

1. Its usage is similar to that of INSTANT ADD COLUMN. You can set `innodb_alter_table_default_algorithm=instant` or specify the `ALGORITHM=instant` keyword when modifying a column.

```
ALTER TABLE t1 MODIFY COLUMN c BIGINT, ALGORITHM=INSTANT;
```

2. The `cdb_instant_modify_column_enabled` switch parameter is added, and the above parameters can take effect only after the switch is on. If the switch is off, the INSTANT MODIFY COLUMN feature will also be disabled.

> Note：
> After the switch is turned off, tables modified with INSTANT MODIFY COLUMN can be used normally.

Restrictions on INSTANT MODIFY COLUMN

- INSTANT MODIFY COLUMN supports modifying only the column type. It supports modifying the `default` attribute of fields but not the `nullable`, `unsigned/signed`, and `charset` attributes.

- Modification is supported only for certain types, and the length can be increased only. Currently, conversions between `char` and `varchar`, between `binary` and `varbinary`, and among `tinyint`, `smallint`, `mediumint`, `int`, and `bigint` are supported.
- One column can be modified with INSTANT MODIFY COLUMN only once, and multiple columns can be modified with it at the same time. After a column is added/modified with INSTANT ADD/MODIFY COLUMN for the first time, it can be modified only in the non-instant manner.
- You can run INSTANT ADD COLUMNS and INSTANT MODIFY COLUMNS separately. You can run INSTANT ADD COLUMNS first and then run INSTANT MODIFY COLUMNS or vice versa. However, you cannot perform INSTANT MODIFY COLUMN on a column that is added with INSTANT ADD COLUMN.
- You cannot modify the column name and column type at the same time. Instead, you can modify the column name first and then column type.
- INSTANT MODIFY COLUMN does not support import/export and index column modification.
- INSTANT MODIFY COLUMN does not support encryption, storage, and redundant format.

Effect Comparison

- INSTANT ADD COLUMN and INSTANT MODIFY COLUMN are performed on the same table with about 50 million data records (12 GB). As can be seen, normal column addition and modification take 2 and 21 minutes respectively, while INSTANT ADD COLUMN and INSTANT MODIFY COLUMN can be done almost instantly, implementing changes within seconds.