

```

AnalyzeOptionListOpt ::=  

( WITH AnalyzeOptionList )?

AnalyzeOptionList ::=  

AnalyzeOption ( ',' AnalyzeOption )*

AnalyzeOption ::=  

( NUM ( 'BUCKETS' | 'TOPN' | ( 'CMSKETCH' ( 'DEPTH' | 'WIDTH' ) ) | 'SAMPLES'  

    ↪ ' ) ) | ( FLOATNUM 'SAMPLERATE' )

TableNameList ::=  

    TableName ( ',' TableName)*

TableName ::=  

    Identifier ( '.' Identifier )?

ColumnNameList ::=  

    Identifier ( ',' Identifier )*

IndexNameList ::=  

    Identifier ( ',' Identifier )*

PartitionNameList ::=  

    Identifier ( ',' Identifier )*

```

11.5.2.15.2 Examples

```

mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT  

    ↪ NOT NULL);  

Query OK, 0 rows affected (0.11 sec)

mysql> INSERT INTO t1 (c1) VALUES (1),(2),(3),(4),(5);  

Query OK, 5 rows affected (0.03 sec)  

Records: 5 Duplicates: 0 Warnings: 0

mysql> ALTER TABLE t1 ADD INDEX (c1);  

Query OK, 0 rows affected (0.30 sec)

mysql> EXPLAIN SELECT * FROM t1 WHERE c1 = 3;  

+--  

→-----+-----+-----+-----+  

→  

| id          | estRows | task      | access object           | operator |

```

```

    ↗ info
+--+
    ↗-----+-----+-----+
    ↗
    ↗
| IndexReader_6      | 10.00 | root      |           | index:
    ↗ IndexRangeScan_5          |
| -IndexRangeScan_5  | 10.00 | cop[tikv] | table:t1, index:c1(c1) | range
    ↗ :[3,3], keep order:false, stats:pseudo |
+--+
    ↗-----+-----+-----+
    ↗
    ↗
2 rows in set (0.00 sec)

mysql> analyze table t1;
Query OK, 0 rows affected (0.13 sec)

mysql> EXPLAIN SELECT * FROM t1 WHERE c1 = 3;
+--+
    ↗-----+-----+-----+
    ↗
    ↗
| id          | estRows | task      | access object      | operator
    ↗ info          |
+--+
    ↗-----+-----+-----+
    ↗
    ↗
| IndexReader_6      | 1.00   | root      |           | index:
    ↗ IndexRangeScan_5          |
| -IndexRangeScan_5  | 1.00   | cop[tikv] | table:t1, index:c1(c1) | range
    ↗ :[3,3], keep order:false |
+--+
    ↗-----+-----+-----+
    ↗
    ↗
2 rows in set (0.00 sec)

```

11.5.2.15.3 MySQL compatibility

TiDB differs from MySQL in **both** the statistics it collects and how it makes use of statistics during query execution. While this statement is syntactically similar to MySQL, the following differences apply:

1. TiDB might not include very recently committed changes when running `ANALYZE TABLE`. After a batch-update of rows, you might need to `sleep(1)` before executing `ANALYZE TABLE` in order for the statistics update to reflect these changes. [#16570](#).

2. `ANALYZE TABLE` takes significantly longer to execute in TiDB than MySQL. This performance difference can be partially mitigated by enabling fast analyze with `SET GLOBAL tidb_enable_fast_analyze=1`. Fast analyze makes use of sampling, leading to less accurate statistics. Its usage is still considered experimental.

MySQL does not support the `ANALYZE INCREMENTAL TABLE` statement. TiDB supports incremental collection of statistics. For detailed usage, refer to [incremental collection](#).

11.5.2.15.4 See also

- [EXPLAIN](#)
- [EXPLAIN ANALYZE](#)

11.5.2.16 BACKUP

This statement is used to perform a distributed backup of the TiDB cluster.

The `BACKUP` statement uses the same engine as the [BR tool](#) does, except that the backup process is driven by TiDB itself rather than a separate BR tool. All benefits and warnings of BR also apply in this statement.

Executing `BACKUP` requires either the `BACKUP_ADMIN` or `SUPER` privilege. Additionally, both the TiDB node executing the backup and all TiKV nodes in the cluster must have read or write permission to the destination. Local storage (storage paths starting with `local://`) is not permitted when [Security Enhanced Mode](#) is enabled.

The `BACKUP` statement is blocked until the entire backup task is finished, failed, or canceled. A long-lasting connection should be prepared for executing `BACKUP`. The task can be canceled using the [KILL TIDB QUERY](#) statement.

Only one `BACKUP` and [RESTORE](#) task can be executed at a time. If a `BACKUP` or `RESTORE` statement is already being executed on the same TiDB server, the new `BACKUP` execution will wait until all previous tasks are finished.

`BACKUP` can only be used with “`tikv`” storage engine. Using `BACKUP` with the “`unistore`” engine will fail.

11.5.2.16.1 Synopsis

```

BackupStmt ::=

    "BACKUP" BRIETables "TO" stringLit BackupOption*

BRIETables ::=

    "DATABASE" ( '*' | DBName ( ',' DBName)* )
    | "TABLE" TableNameList

BackupOption ::=
```

```

|R "RATE_LIMIT" '='? LengthNum "MB" '/' "SECOND"
| "CONCURRENCY" '='? LengthNum
| "CHECKSUM" '='? Boolean
| "SEND_CREDENTIALS_TO_TIKV" '='? Boolean
| "LAST_BACKUP" '='? BackupTSO
| "SNAPSHOT" '='? ( BackupTSO | LengthNum TimestampUnit "AGO" )

Boolean ::=

NUM | "TRUE" | "FALSE"

BackupTSO ::=

LengthNum | stringLit

```

11.5.2.16.2 Examples

Back up databases

```
BACKUP DATABASE `test` TO 'local:///mnt/backup/2020/04/';
```

Destination	Size	BackupTS	Queue Time
Execution Time			
local:///mnt/backup/2020/04/	248665063	416099531454472	2020-04-12
23:09:48	2020-04-12 23:09:48		

1 row in set (58.453 sec)

In the example above, the `test` database is backed up into the local filesystem. The data is saved as SST files in the `/mnt/backup/2020/04/` directories distributed among all TiDB and TiKV nodes.

The first row of the result above is described as follows:

Column	Description
Destination	The destination URL

Column	Description
Size	The total size of the backup archive, in bytes
BackupTS	The TSO of the snapshot when the backup is created (useful for incremental backup)
Queue → Time	The timestamp (in current time zone) when the BACKUP task is queued.
Execution → Time	The timestamp (in current time zone) when the BACKUP task starts to run.

Back up tables

```
BACKUP TABLE `test`.`sbtest01` TO 'local:///mnt/backup/sbtest01/' ;
```

```
BACKUP TABLE sbtest02, sbtest03, sbtest04 TO 'local:///mnt/backup/sbtest/' ;
```

Back up the entire cluster

```
BACKUP DATABASE * TO 'local:///mnt/backup/full/' ;
```

Note that the system tables (`mysql.*`, `INFORMATION_SCHEMA.*`, `PERFORMANCE_SCHEMA.*`, ... will not be included into the backup.

External storages

BR supports backing up data to S3 or GCS:

```
BACKUP DATABASE `test` TO 's3://example-bucket-2020/backup-05/?region=us-
    ↪ west-2&access-key={YOUR_ACCESS_KEY}&secret-access-key={
    ↪ YOUR_SECRET_KEY}' ;
```

The URL syntax is further explained in [External Storages](#).

When running on cloud environment where credentials should not be distributed, set the `SEND_CREDENTIALS_TO_TIKV` option to `FALSE`:

```
BACKUP DATABASE `test` TO 's3://example-bucket-2020/backup-05/?region=us-
    ↪ west-2'
    SEND_CREDENTIALS_TO_TIKV = FALSE;
```

Performance fine-tuning

Use `RATE_LIMIT` to limit the average upload speed per TiKV node to reduce network bandwidth.

By default, every TiKV node would run 4 backup threads. This value can be adjusted with the `CONCURRENCY` option.

Before backup is completed, `BACKUP` would perform a checksum against the data on the cluster to verify correctness. This step can be disabled with the `CHECKSUM` option if you are confident that this is unnecessary.

```
BACKUP DATABASE `test` TO 's3://example-bucket-2020/backup-06/'
    RATE_LIMIT = 120 MB/SECOND
    CONCURRENCY = 8
    CHECKSUM = FALSE;
```

Snapshot

Specify a timestamp, TSO or relative time to backup historical data.

```
-- relative time
BACKUP DATABASE `test` TO 'local:///mnt/backup/hist01'
    SNAPSHOT = 36 HOUR AGO;

-- timestamp (in current time zone)
BACKUP DATABASE `test` TO 'local:///mnt/backup/hist02'
    SNAPSHOT = '2020-04-01 12:00:00';

-- timestamp oracle
BACKUP DATABASE `test` TO 'local:///mnt/backup/hist03'
    SNAPSHOT = 415685305958400;
```

The supported units for relative time are:

- MICROSECOND
- SECOND
- MINUTE
- HOUR
- DAY
- WEEK

Note that, following SQL standard, the units are always singular.

Incremental backup

Supply the `LAST_BACKUP` option to only backup the changes between the last backup to the current snapshot.

```
-- timestamp (in current time zone)
BACKUP DATABASE `test` TO 'local:///mnt/backup/hist02'
    LAST_BACKUP = '2020-04-01 12:00:00';

-- timestamp oracle
BACKUP DATABASE `test` TO 'local:///mnt/backup/hist03'
    LAST_BACKUP = 415685305958400;
```

11.5.2.16.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.16.4 See also

- [RESTORE](#)
- [SHOW BACKUPS](#)

11.5.2.17 BEGIN

This statement starts a new transaction inside of TiDB. It is similar to the statements `START TRANSACTION` and `SET autocommit=0`.

In the absence of a `BEGIN` statement, every statement will by default autocommit in its own transaction. This behavior ensures MySQL compatibility.

11.5.2.17.1 Synopsis

```
BeginTransactionStmt ::= 
    'BEGIN' ( 'PESSIMISTIC' | 'OPTIMISTIC' )?
| 'START' 'TRANSACTION' ( 'READ' ( 'WRITE' | 'ONLY' ( 'WITH' 'TIMESTAMP' '
    ↪ BOUND' TimestampBound )? ) | 'WITH' 'CONSISTENT' 'SNAPSHOT' )?
```

11.5.2.17.2 Examples

```
mysql> CREATE TABLE t1 (a int NOT NULL PRIMARY KEY);
Query OK, 0 rows affected (0.12 sec)

mysql> BEGIN;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO t1 VALUES (1);
Query OK, 1 row affected (0.00 sec)

mysql> COMMIT;
Query OK, 0 rows affected (0.01 sec)
```

11.5.2.17.3 MySQL compatibility

TiDB supports the syntax extension of BEGIN PESSIMISTIC or BEGIN OPTIMISTIC. This enables you to override the default transactional model for your transaction.

11.5.2.17.4 See also

- [COMMIT](#)
- [ROLLBACK](#)
- [START TRANSACTION](#)
- [TiDB optimistic transaction model](#)
- [TiDB pessimistic transaction mode](#)

11.5.2.18 CHANGE COLUMN

The ALTER TABLE.. CHANGE COLUMN statement changes a column on an existing table. The change can include both renaming the column, and changing the data type to a compatible type.

Since v5.1.0, TiDB has supported changing the Reorg data type, including but not limited to:

- Changing VARCHAR to BIGINT
- Modifying the DECIMAL precision
- Compressing the length of VARCHAR(10) to VARCHAR(5)

11.5.2.18.1 Synopsis

```
AlterTableStmt ::=

'ALTER' IgnoreOptional 'TABLE' TableName ( AlterTableSpecListOpt
    ↪ AlterTablePartitionOpt | 'ANALYZE' 'PARTITION' PartitionNameList (
    ↪ 'INDEX' IndexNameList )? AnalyzeOptionListOpt )
```

```

AlterTableSpec ::=

    TableOptionList
| 'SET' 'TIFLASH' 'REPLICA' LengthNum LocationLabelList
| 'CONVERT' 'TO' CharsetKw ( CharsetName | 'DEFAULT' ) OptCollate
| 'ADD' ( ColumnKeywordOpt IfNotExists ( ColumnDef ColumnPosition | '('
    ↪ TableElementList ')' ) | Constraint | 'PARTITION' IfNotExists
    ↪ NoWriteToBinLogAliasOpt ( PartitionDefinitionListOpt | 'PARTITIONS'
    ↪ NUM ) )
| ( ( 'CHECK' | 'TRUNCATE' ) 'PARTITION' | ( 'OPTIMIZE' | 'REPAIR' | '
    ↪ REBUILD' ) 'PARTITION' NoWriteToBinLogAliasOpt )
    ↪ AllOrPartitionNameList
| 'COALESCE' 'PARTITION' NoWriteToBinLogAliasOpt NUM
| 'DROP' ( ColumnKeywordOpt IfExists ColumnName RestrictOrCascadeOpt | '
    ↪ PRIMARY' 'KEY' | 'PARTITION' IfExists PartitionNameList | (
        ↪ KeyOrIndex IfExists | 'CHECK' ) Identifier | 'FOREIGN' 'KEY' IfExists
        ↪ Symbol )
| 'EXCHANGE' 'PARTITION' Identifier 'WITH' 'TABLE' TableName
    ↪ WithValidationOpt
| ( 'IMPORT' | 'DISCARD' ) ( 'PARTITION' AllOrPartitionNameList )? ' '
    ↪ TABLESPACE'
| 'REORGANIZE' 'PARTITION' NoWriteToBinLogAliasOpt
    ↪ ReorganizePartitionRuleOpt
| 'ORDER' 'BY' AlterOrderItem ( ',' AlterOrderItem )*
| ( 'DISABLE' | 'ENABLE' ) 'KEYS'
| ( 'MODIFY' ColumnKeywordOpt IfExists | 'CHANGE' ColumnKeywordOpt
    ↪ IfExists ColumnName ) ColumnDef ColumnPosition
| 'ALTER' ( ColumnKeywordOpt ColumnName ( 'SET' 'DEFAULT' ( SignedLiteral
    ↪ | '(' Expression ')' ) | 'DROP' 'DEFAULT' ) | 'CHECK' Identifier
    ↪ EnforcedOrNot | 'INDEX' Identifier IndexInvisible )
| 'RENAME' ( ( 'COLUMN' | KeyOrIndex ) Identifier 'TO' Identifier | ( 'TO'
    ↪ | '='? | 'AS' ) TableName )
| LockClause
| AlgorithmClause
| 'FORCE'
| ( 'WITH' | 'WITHOUT' ) 'VALIDATION'
| 'SECONDARY_LOAD'
| 'SECONDARY_UNLOAD'

ColumnName ::=

    Identifier ( '.' Identifier ( '.' Identifier )? )?

ColumnDef ::=

    ColumnName ( Type | 'SERIAL' ) ColumnOptionListOpt

```

```
ColumnPosition ::=  
  ('FIRST' | 'AFTER' ColumnName)?
```

11.5.2.18.2 Examples

```
CREATE TABLE t1 (id int not null primary key AUTO_INCREMENT, col1 INT);
```

```
Query OK, 0 rows affected (0.11 sec)
```

```
INSERT INTO t1 (col1) VALUES (1),(2),(3),(4),(5);
```

```
Query OK, 5 rows affected (0.02 sec)
```

```
Records: 5 Duplicates: 0 Warnings: 0
```

```
ALTER TABLE t1 CHANGE col1 col2 INT;
```

```
Query OK, 0 rows affected (0.09 sec)
```

```
ALTER TABLE t1 CHANGE col2 col3 BIGINT, ALGORITHM=INSTANT;
```

```
Query OK, 0 rows affected (0.08 sec)
```

```
ALTER TABLE t1 CHANGE col3 col4 BIGINT, CHANGE id id2 INT NOT NULL;
```

```
ERROR 1105 (HY000): can't run multi schema change
```

```
CREATE TABLE t (a int primary key);  
ALTER TABLE t CHANGE COLUMN a a VARCHAR(10);
```

```
ERROR 8200 (HY000): Unsupported modify column: column has primary key flag
```

```
CREATE TABLE t (c1 INT, c2 INT, c3 INT) partition by range columns(c1) (  
    ↪ partition p0 values less than (10), partition p1 values less than (  
    ↪ maxvalue));  
ALTER TABLE t CHANGE COLUMN c1 c1 DATETIME;
```

```
ERROR 8200 (HY000): Unsupported modify column: table is partition table
```

```
CREATE TABLE t (a INT, b INT as (a+1));  
ALTER TABLE t CHANGE COLUMN b b VARCHAR(10);
```

```
ERROR 8200 (HY000): Unsupported modify column: column is generated
```

```
CREATE TABLE t (a DECIMAL(13, 7));
ALTER TABLE t CHANGE COLUMN a a DATETIME;
```

ERROR 8200 (HY000): Unsupported modify column: change from original type
 ↪ decimal(13,7) to datetime is currently unsupported yet

11.5.2.18.3 MySQL compatibility

- Making multiple changes in a single `ALTER TABLE` statement is currently not supported.
- Changes of **Reorg-Data** types on primary key columns are not supported.
- Changes of column types on partitioned tables are not supported.
- Changes of column types on generated columns are not supported.
- Changes of some data types (for example, some TIME, Bit, Set, Enum, and JSON types) are not supported due to the compatibility issues of the `CAST` function's behavior between TiDB and MySQL.

11.5.2.18.4 See also

- [CREATE TABLE](#)
- [SHOW CREATE TABLE](#)
- [ADD COLUMN](#)
- [DROP COLUMN](#)
- [MODIFY COLUMN](#)

11.5.2.19 COMMIT

This statement commits a transaction inside of the TiDB server.

In the absence of a `BEGIN` or `START TRANSACTION` statement, the default behavior of TiDB is that every statement will be its own transaction and autocommit. This behavior ensures MySQL compatibility.

11.5.2.19.1 Synopsis

```
CommitStmt ::=

    'COMMIT' CompletionTypeWithinTransaction?

CompletionTypeWithinTransaction ::=

    'AND' ( 'CHAIN' ( 'NO' 'RELEASE' )? | 'NO' 'CHAIN' ( 'NO'? 'RELEASE' )?
        ↪ )
    | 'NO'? 'RELEASE'
```

11.5.2.19.2 Examples

```
mysql> CREATE TABLE t1 (a int NOT NULL PRIMARY KEY);
Query OK, 0 rows affected (0.12 sec)

mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO t1 VALUES (1);
Query OK, 1 row affected (0.00 sec)

mysql> COMMIT;
Query OK, 0 rows affected (0.01 sec)
```

11.5.2.19.3 MySQL compatibility

- By default, TiDB 3.0.8 and later versions use [Pessimistic Locking](#). When using [Optimistic Locking](#), it is important to consider that a COMMIT statement might fail because rows have been modified by another transaction.
- When Optimistic Locking is enabled, UNIQUE and PRIMARY KEY constraint checks are deferred until statement commit. This results in additional situations where a COMMIT statement might fail. This behavior can be changed by setting `tidb_constraint_check_in_place=TRUE`.
- TiDB parses but ignores the syntax ROLLBACK AND [NO] RELEASE. This functionality is used in MySQL to disconnect the client session immediately after committing the transaction. In TiDB, it is recommended to instead use the `mysql_close()` functionality of your client driver.
- TiDB parses but ignores the syntax ROLLBACK AND [NO] CHAIN. This functionality is used in MySQL to immediately start a new transaction with the same isolation level while the current transaction is being committed. In TiDB, it is recommended to instead start a new transaction.

11.5.2.19.4 See also

- [START TRANSACTION](#)
- [ROLLBACK](#)
- [BEGIN](#)
- [Lazy checking of constraints](#)

11.5.2.20 CHANGE DRAINER

The `CHANGE DRAINER` statement modifies the status information for Drainer in the cluster.

Tip:

Drainer's state is automatically reported to PD while running. Only when Drainer is under abnormal circumstances and its state is inconsistent with the state information stored in PD, you can use the `CHANGE DRAINER` statement to modify the state information stored in PD.

11.5.2.20.1 Examples

```
SHOW DRAINER STATUS;
```

NodeID	Address	State	Max_Commit_Ts	Update_Time
drainer1	127.0.0.3:8249	Online	408553768673342532	2019-04-30 00:00:03
drainer2	127.0.0.4:8249	Online	408553768673345531	2019-05-01 00:00:04

2 rows in set (0.00 sec)

It can be seen that drainer1's state has not been updated for more than a day, the Drainer is in an abnormal state, but the State remains `Online`. After using `CHANGE DRAINER`, the Drainer's State is changed to 'paused':

```
CHANGE DRAINER TO NODE_STATE = 'paused' FOR NODE_ID 'drainer1';
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
SHOW DRAINER STATUS;
```

NodeID	Address	State	Max_Commit_Ts	Update_Time
--------	---------	-------	---------------	-------------

NodeID	Address	State	Max_Commit_Ts	Update_Time
drainer1	127.0.0.3:8249	Paused	408553768673342532	2019-04-30 00:00:03
drainer2	127.0.0.4:8249	Online	408553768673345531	2019-05-01 00:00:04
2 rows in set (0.00 sec)				

11.5.2.20.2 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.20.3 See also

- [SHOW PUMP STATUS](#)
- [SHOW DRAINER STATUS](#)
- [CHANGE PUMP STATUS](#)

11.5.2.21 CHANGE PUMP

The `CHANGE PUMP` statement modifies the status information for Pump in the cluster.

Tip:

Pump's state is automatically reported to PD while running. Only when Pump is under abnormal circumstances and its state is inconsistent with the state information stored in PD, you can use the `CHANGE PUMP` statement to modify the state information stored in PD.

11.5.2.21.1 Examples

```
SHOW PUMP STATUS;
```

```
+--+
| NodeID | Address | State | Max_Commit_Ts | Update_Time |
+--+
| pump1 | 127.0.0.1:8250 | Online | 408553768673342237 | 2019-04-30 00:00:01
| pump2 | 127.0.0.2:8250 | Online | 408553768673342335 | 2019-05-01 00:00:02
+--+
| pump1 | 127.0.0.1:8250 | Paused | 408553768673342237 | 2019-04-30 00:00:01
| pump2 | 127.0.0.2:8250 | Online | 408553768673342335 | 2019-05-01 00:00:02
+--+
2 rows in set (0.00 sec)
```

It can be seen that `pump1`'s state has not been updated for more than a day, the Pump is in an abnormal state, but the `State` remains `Online`. After using `CHANGE PUMP`, the Pump's `State` is changed to 'paused' :

```
CHANGE PUMP TO NODE_STATE ='paused' FOR NODE_ID 'pump1';
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
SHOW PUMP STATUS;
```

```
+--+
| NodeID | Address | State | Max_Commit_Ts | Update_Time |
+--+
| pump1 | 127.0.0.1:8250 | Paused | 408553768673342237 | 2019-04-30 00:00:01
| pump2 | 127.0.0.2:8250 | Online | 408553768673342335 | 2019-05-01 00:00:02
+--+
```

```
+--+
→ ----- / ----- / ----- / ----- / -----
→
2 rows in set (0.00 sec)
```

11.5.2.21.2 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.21.3 See also

- [SHOW PUMP STATUS](#)
- [SHOW DRAINER STATUS](#)
- [CHANGE DRAINER STATUS](#)

11.5.2.22 CREATE [GLOBAL|SESSION] BINDING

This statement creates a new execution plan binding in TiDB. Binding can be used to inject a hint into a statement without requiring changes to the underlying query.

A `BINDING` can be on either a `GLOBAL` or `SESSION` basis. The default is `SESSION`.

The bound SQL statement is parameterized and stored in the system table. When a SQL query is processed, as long as the parameterized SQL statement and a bound one in the system table are consistent and the system variable `tidb_use_plan_baselines` is set to `ON` (default), the corresponding optimizer hint is available. If multiple execution plans are available, the optimizer chooses to bind the plan with the least cost.

11.5.2.22.1 Synopsis

```
CreateBindingStmt ::= 'CREATE' GlobalScope 'BINDING' 'FOR' BindableStmt 'USING' BindableStmt

GlobalScope ::= ('GLOBAL' | 'SESSION')?

BindableStmt ::= (SelectStmt | UpdateStmt | InsertIntoStmt | ReplaceIntoStmt |
                  → DeleteStmt)
```

11.5.2.22.2 Examples

```

mysql> CREATE TABLE t1 (
    -> id INT NOT NULL PRIMARY KEY auto_increment,
    -> b INT NOT NULL,
    -> pad VARBINARY(255),
    -> INDEX(b)
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    ↪ FROM dual;
Query OK, 1 row affected (0.01 sec)
Records: 1 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 1 row affected (0.00 sec)
Records: 1 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 8 rows affected (0.00 sec)
Records: 8 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 1000 rows affected (0.04 sec)
Records: 1000 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 100000 rows affected (1.74 sec)
Records: 100000 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 100000 rows affected (2.15 sec)
Records: 100000 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 100000 rows affected (2.64 sec)
Records: 100000 Duplicates: 0 Warnings: 0

```

```

mysql> SELECT SLEEP(1);
+-----+
| SLEEP(1) |
+-----+
|      0   |
+-----+
1 row in set (1.00 sec)

mysql> ANALYZE TABLE t1;
Query OK, 0 rows affected (1.33 sec)

mysql> EXPLAIN ANALYZE SELECT * FROM t1 WHERE b = 123;
+--+
| id           | estRows | actRows | task | access object
|             | execution info
|             |           |           | operator info
|             | memory    | disk    |
+--+
| IndexLookUp_10 | 583.00 | 297    | root  | 
|               | time:10.545072ms, loops:2, rpc num: 1, rpc time
|               | :398.359µs, proc keys:297 | 109.1484375 KB | N/
|               | A |
| -IndexRangeScan_8(Build) | 583.00 | 297    | cop[tikv] | table:t1, index
|   :b(b) | time:0s, loops:4
|   range:[123,123], keep order:false | N/A     | N/A |
| -TableRowIDScan_9(Probe) | 583.00 | 297    | cop[tikv] | table:t1
|   | time:12ms, loops:4
|   | keep order:false
|   | N/A     | N/A |
+--+
|                                     |
|                                     |
3 rows in set (0.02 sec)

mysql> CREATE SESSION BINDING FOR
->   SELECT * FROM t1 WHERE b = 123
->   USING
->   SELECT * FROM t1 IGNORE INDEX (b) WHERE b = 123;
Query OK, 0 rows affected (0.00 sec)

mysql> EXPLAIN ANALYZE SELECT * FROM t1 WHERE b = 123;

```

```
+--+
| id           | estRows | actRows | task      | access object |
|   ↗ execution info
|   ↗ operator info | memory       | disk |
+--+
| TableReader_7    | 583.00  | 297    | root      |               | time
|   ↗ :222.32506ms, loops:2, rpc num: 1, rpc time:222.078952ms, proc keys
|   ↗ :301010 | data:Selection_6 | 88.6640625 KB | N/A |
| -Selection_6    | 583.00  | 297    | cop[tikv] |               | time
|   ↗ :224ms, loops:298
|   ↗ test.t1.b, 123 | N/A        | N/A |
| -TableFullScan_5 | 301010.00 | 301010 | cop[tikv] | table:t1 | time
|   ↗ :220ms, loops:298
|   ↗ keep order:false | N/A        | N/A |
+--+
| ↗
| ↗
3 rows in set (0.22 sec)

mysql> SHOW SESSION BINDINGS\G
***** 1. row *****
Original_sql: select * from t1 where b = ?
Bind_sql: SELECT * FROM t1 IGNORE INDEX (b) WHERE b = 123
Default_db: test
Status: using
Create_time: 2020-05-22 14:38:03.456
Update_time: 2020-05-22 14:38:03.456
Charset: utf8mb4
Collation: utf8mb4_0900_ai_ci
1 row in set (0.00 sec)

mysql> DROP SESSION BINDING FOR SELECT * FROM t1 WHERE b = 123;
Query OK, 0 rows affected (0.00 sec)

mysql> EXPLAIN ANALYZE SELECT * FROM t1 WHERE b = 123;
+--+
| id           | estRows | actRows | task      | access object |
|   ↗ execution info
|   ↗ operator info | memory       | disk |
+--+
```

```
+--+
| IndexLookUp_10           | 583.00 | 297    | root      |
|                         | time:5.31206ms, loops:2, rpc num: 1, rpc time |
|                         | :665.927µs, proc keys:297 | 109.1484375 KB | N
|                         | /A | | | |
| -IndexRangeScan_8(Build) | 583.00 | 297    | cop[tikv] | table:t1, index |
|                         | :b(b) | time:0s, loops:4 |
|                         | range:[123,123], keep order:false | N/A      | N/A |
| -TableRowIDScan_9(Probe) | 583.00 | 297    | cop[tikv] | table:t1 |
|                         | time:0s, loops:4 |
|                         | N/A      | N/A | keep order:false
+--+
| 3 rows in set (0.01 sec)
```

11.5.2.22.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.22.4 See also

- [DROP \[GLOBAL|SESSION\] BINDING](#)
- [SHOW \[GLOBAL|SESSION\] BINDINGS](#)
- [ANALYZE TABLE](#)
- [Optimizer Hints](#)
- [SQL Plan Management](#)

11.5.2.23 CREATE DATABASE

This statement creates a new database in TiDB. The MySQL terminology for ‘database’ most closely maps to a schema in the SQL standard.

11.5.2.23.1 Synopsis

```
CreateDatabaseStmt ::=  
  'CREATE' 'DATABASE' IfNotExists DBName DatabaseOptionListOpt  
  
IfNotExists ::=  
  ('IF' 'NOT' 'EXISTS')?
```

```

DBName ::= Identifier
DatabaseOptionListOpt ::= DatabaseOptionList?

```

11.5.2.23.2 Syntax

The `CREATE DATABASE` statement is used to create a database, and to specify the default properties of the database, such as the default character set and collation. `CREATE SCHEMA` is a synonym for `CREATE DATABASE`.

```

CREATE {DATABASE | SCHEMA} [IF NOT EXISTS] db_name
      [create_specification] ...

create_specification:
    [DEFAULT] CHARACTER SET [=] charset_name
    | [DEFAULT] COLLATE [=] collation_name

```

If you create an existing database and does not specify `IF NOT EXISTS`, an error is displayed.

The `create_specification` option is used to specify the specific `CHARACTER SET` and `COLLATE` in the database. Currently, TiDB only supports some of the character sets and collations. For details, see [Character Set and Collation Support](#).

11.5.2.23.3 Examples

```

mysql> CREATE DATABASE mynewdatabase;
Query OK, 0 rows affected (0.09 sec)

mysql> USE mynewdatabase;
Database changed
mysql> CREATE TABLE t1 (a int);
Query OK, 0 rows affected (0.11 sec)

mysql> SHOW TABLES;
+-----+
| Tables_in_mynewdatabase |
+-----+
| t1                         |
+-----+
1 row in set (0.00 sec)

```

11.5.2.23.4 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be [reported via an issue](#) on GitHub.

11.5.2.23.5 See also

- [USE](#)
- [ALTER DATABASE](#)
- [DROP DATABASE](#)
- [SHOW DATABASES](#)

11.5.2.24 CREATE INDEX

This statement adds a new index to an existing table. It is an alternative syntax to `ALTER TABLE ... ADD INDEX`, and included for MySQL compatibility.

11.5.2.24.1 Synopsis

```

CreateIndexStmt ::=

  'CREATE' IndexKeyTypeOpt 'INDEX' IfNotExists Identifier IndexTypeOpt 'ON'
    ↪ ' TableName '(' IndexPartSpecificationList ')' IndexOptionList
    ↪ IndexLockAndAlgorithmOpt

IndexKeyTypeOpt ::=

  ( 'UNIQUE' | 'SPATIAL' | 'FULLTEXT' )?

IfNotExists ::=

  ( 'IF' 'NOT' 'EXISTS' )?

IndexTypeOpt ::=

  IndexType?

IndexPartSpecificationList ::=

  IndexPartSpecification ( ',' IndexPartSpecification )*

IndexOptionList ::=

  IndexOption*

IndexLockAndAlgorithmOpt ::=

  ( LockClause AlgorithmClause? | AlgorithmClause LockClause? )?

IndexType ::=

  ( 'USING' | 'TYPE' ) IndexTypeName

```

```

IndexPartSpecification ::= 
    ( ColumnName OptFieldLen | '(' Expression ')' ) Order

IndexOption ::= 
    'KEY_BLOCK_SIZE' '='? LengthNum
| IndexType
| 'WITH' 'PARSER' Identifier
| 'COMMENT' stringLit
| IndexInvisible

IndexTypeName ::= 
    'BTREE'
| 'HASH'
| 'RTREE'

ColumnName ::= 
    Identifier ( '.' Identifier ( '.' Identifier )? )?

OptFieldLen ::= 
    FieldLen?

IndexNameList ::= 
    ( Identifier | 'PRIMARY' )? ( ',' ( Identifier | 'PRIMARY' ) )* 

KeyOrIndex ::= 
    'Key' | 'Index'

```

11.5.2.24.2 Examples

```

mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT
   ↪ NOT NULL);
Query OK, 0 rows affected (0.10 sec)

mysql> INSERT INTO t1 (c1) VALUES (1),(2),(3),(4),(5);
Query OK, 5 rows affected (0.02 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> EXPLAIN SELECT * FROM t1 WHERE c1 = 3;
+---+
   ↪ -----+-----+-----+
   ↪
| id          | estRows | task      | access object | operator info
   ↪
+---+

```

```

    ↵ +-----+-----+-----+
    ↵ |       |
    ↵ | TableReader_7      | 10.00   | root      |           | data:Selection_6
    ↵ |       |
    ↵ | -Selection_6       | 10.00   | cop[tikv] |           | eq(test.t1.c1,
    ↵ |       |           |            |           |   3)
    ↵ |   -TableFullScan_5 | 10000.00 | cop[tikv] | table:t1 | keep order:false
    ↵ |                   |           |            |           | , stats:pseudo |
+--+
    ↵ +-----+-----+-----+
    ↵ |
    ↵ 3 rows in set (0.00 sec)

mysql> CREATE INDEX c1 ON t1 (c1);
Query OK, 0 rows affected (0.30 sec)

mysql> EXPLAIN SELECT * FROM t1 WHERE c1 = 3;
+-
    ↵ +-----+-----+-----+
    ↵ |       |
    ↵ | id      | estRows | task      | access object      | operator
    ↵ | info    |          |           |                  |
+-
    ↵ +-----+-----+-----+
    ↵ |
    ↵ | IndexReader_6      | 10.00   | root      |           | index:
    ↵ |                   |          |           |           |
    ↵ | -IndexRangeScan_5  | 10.00   | cop[tikv] | table:t1, index:c1(c1) | range
    ↵ |                   |          |            |           |   :[3,3], keep order:false, stats:pseudo |
+-
    ↵ +-----+-----+-----+
    ↵ |
    ↵ 2 rows in set (0.00 sec)

mysql> ALTER TABLE t1 DROP INDEX c1;
Query OK, 0 rows affected (0.30 sec)

mysql> CREATE UNIQUE INDEX c1 ON t1 (c1);
Query OK, 0 rows affected (0.31 sec)

```

11.5.2.24.3 Expression index

In some scenarios, the filtering condition of a query is based on a certain expression. In these scenarios, the query performance is relatively poor because ordinary indexes cannot

take effect, the query can only be executed by scanning the entire table. The expression index is a type of special index that can be created on an expression. Once an expression index is created, TiDB can use the index for the expression-based query, which significantly improves the query performance.

For example, if you want to create an index based on `lower(col1)`, execute the following SQL statement:

```
CREATE INDEX idx1 ON t1 ((lower(col1)));
```

Or you can execute the following equivalent statement:

```
ALTER TABLE t1 ADD INDEX idx1((lower(col1)));
```

You can also specify the expression index when you create the table:

```
CREATE TABLE t1(col1 char(10), col2 char(10), key index((lower(col1))));
```

Note

The expression in an expression index must be surrounded by '(' and ')'. Otherwise, a syntax error is reported.

You can drop an expression index in the same way as dropping an ordinary index:

```
DROP INDEX idx1 ON t1;
```

Note:

Expression index involves various kinds of expressions. To ensure correctness, only some fully tested functions are allowed for creating an expression index. This means that only these functions are allowed in expressions in a production environment. You can get these functions by querying `tidb_allow_function_for_expression_index` variable. In future versions, more functions might be added to the list.

```
mysql> select @@tidb_allow_function_for_expression_index;
+-----+
| @@tidb_allow_function_for_expression_index |
+-----+
| lower, md5, reverse, upper, vitess_hash |
+-----+
1 row in set (0.00 sec)
```

TiDB configuration file

index true

→

When the expression in a query statement matches the expression in an expression index, the optimizer can choose the expression index for the query. In some cases, the optimizer might not choose an expression index depending on statistics. In this situation, you can force the optimizer to select an expression index by using optimizer hints.

In the following examples, suppose that you create the expression index `idx` on the expression `lower(col1)`:

If the results of the query statement are the same expressions, the expression index applies. Take the following statement as an example:

```
SELECT lower(col1) FROM t;
```

If the same expression is included in the filtering conditions, the expression index applies. Take the following statements as an example:

```
SELECT * FROM t WHERE lower(col1) = "a";
SELECT * FROM t WHERE lower(col1) > "a";
SELECT * FROM t WHERE lower(col1) BETWEEN "a" AND "b";
SELECT * FROM t WHERE lower(col1) IN ("a", "b");
SELECT * FROM t WHERE lower(col1) > "a" AND lower(col1) < "b";
SELECT * FROM t WHERE lower(col1) > "b" OR lower(col1) < "a";
```

When the queries are sorted by the same expression, the expression index applies. Take the following statement as an example:

```
SELECT * FROM t ORDER BY lower(col1);
```

If the same expression is included in the aggregate (GROUP BY) functions, the expression index applies. Take the following statements as an example:

```
SELECT max(lower(col1)) FROM t;
SELECT min(col1) FROM t GROUP BY lower(col1);
```

To see the expression corresponding to the expression index, execute `SHOW INDEX` →, or check the system tables `information_schema.tidb_indexes` and the table `information_schema.STATISTICS`. The `Expression` column in the output indicates the corresponded expression. For the non-expression indexes, the column shows `NULL`.

The cost of maintaining an expression index is higher than that of maintaining other indexes, because the value of the expression needs to be calculated whenever a row is inserted or updated. The value of the expression is already stored in the index, so this value does not require recalculation when the optimizer selects the expression index.

Therefore, when the query performance outweighs the insert and update performance, you can consider indexing the expressions.

Expression indexes have the same syntax and limitations as in MySQL. They are implemented by creating indexes on generated virtual columns that are invisible, so the supported expressions inherit all [limitations of virtual generated columns](#).

11.5.2.24.4 Invisible index

Invisible indexes are indexes that are ignored by the query optimizer:

```
CREATE TABLE t1 (c1 INT, c2 INT, UNIQUE(c2));
CREATE UNIQUE INDEX c1 ON t1 (c1) INVISIBLE;
```

For details, see [ALTER INDEX](#).

11.5.2.24.5 Associated system variables

The system variables associated with the `CREATE INDEX` statement are `tidb_ddl_reorg_worker_cnt`, `tidb_ddl_reorg_batch_size`, `tidb_enable_auto_increment_in_generated`, and `tidb_ddl_reorg_priority`. Refer to [system variables](#) for details.

11.5.2.24.6 MySQL compatibility

- `FULLTEXT`, `HASH` and `SPATIAL` indexes are not supported.
- Descending indexes are not supported (similar to MySQL 5.7).
- Adding the primary key of the `CLUSTERED` type to a table is not supported. For more details about the primary key of the `CLUSTERED` type, refer to [clustered index](#).
- Expression indexes are incompatible with views. When a query is executed using a view, the expression index cannot be used at the same time.
- Expression indexes have compatibility issues with bindings. When the expression of an expression index has a constant, the binding created for the corresponding query expands its scope. For example, suppose that the expression in the expression index is `a+1`, and the corresponding query condition is `a+1 > 2`. In this case, the created binding is `a+? > ?`, which means that the query with the condition such as `a+2 > 2` is also forced to use the expression index and results in a poor execution plan. In addition, this also affects the baseline capturing and baseline evolution in SQL Plan Management (SPM).

11.5.2.24.7 See also

- [Index Selection](#)
- [Wrong Index Solution](#)
- [ADD INDEX](#)
- [DROP INDEX](#)
- [RENAME INDEX](#)
- [ALTER INDEX](#)
- [ADD COLUMN](#)
- [CREATE TABLE](#)
- [EXPLAIN](#)

11.5.2.25 CREATE PLACEMENT POLICY

Warning:

Placement Rules in SQL is an experimental feature. The syntax might change before its GA, and there might also be bugs.

If you understand the risks, you can enable this experiment feature by executing `SET GLOBAL tidb_enable_alter_placement = 1;`.

CREATE PLACEMENT POLICY is used to create a named placement policy that can later be assigned to tables, partitions, or database schemas.

11.5.2.25.1 Synopsis

```

CreatePolicyStmt ::=

    "CREATE" "PLACEMENT" "POLICY" IfNotExists PolicyName PlacementOptionList

PolicyName ::=

    Identifier

PlacementOptionList ::=

    DirectPlacementOption
    | PlacementOptionList DirectPlacementOption
    | PlacementOptionList ',' DirectPlacementOption

DirectPlacementOption ::=

    "PRIMARY_REGION" EqOpt stringLit
    | "REGIONS" EqOpt stringLit
    | "FOLLOWERS" EqOpt LengthNum
    | "VOTERS" EqOpt LengthNum
    | "LEARNERS" EqOpt LengthNum
    | "SCHEDULE" EqOpt stringLit
    | "CONSTRAINTS" EqOpt stringLit
    | "LEADER_CONSTRAINTS" EqOpt stringLit
    | "FOLLOWER_CONSTRAINTS" EqOpt stringLit
    | "VOTER_CONSTRAINTS" EqOpt stringLit
    | "LEARNER_CONSTRAINTS" EqOpt stringLit

```

11.5.2.25.2 Examples

Note:

To know which regions are available in your cluster, see [SHOW PLACEMENT](#)
 ↪ [LABELS](#).

If you do not see any available regions, your TiKV installation might not have labels set correctly.

```

CREATE PLACEMENT POLICY p1 PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us
    ↪ -west-1" FOLLOWERS=4;
CREATE TABLE t1 (a INT) PLACEMENT POLICY=p1;

```

```
SHOW CREATE PLACEMENT POLICY p1;
```

```
Query OK, 0 rows affected (0.08 sec)
```

```
Query OK, 0 rows affected (0.10 sec)
```

```
+-----+
   ↗
| Policy | Create Policy
   ↗
   ↗ |
+-----+
   ↗
| p1    | CREATE PLACEMENT POLICY `p1` PRIMARY_REGION="us-east-1" REGIONS=
   ↗ us-east-1,us-west-1" FOLLOWERS=4 |
+-----+
   ↗
1 row in set (0.00 sec)
```

11.5.2.25.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.25.4 See also

- [Placement Rules in SQL](#)
- [SHOW PLACEMENT](#)
- [ALTER PLACEMENT POLICY](#)
- [DROP PLACEMENT POLICY](#)

11.5.2.26 CREATE ROLE

This statement creates a new role, which can be assigned to users as part of role-based access control.

11.5.2.26.1 Synopsis

```
CreateRoleStmt ::=  
  'CREATE' 'ROLE' IfNotExists RoleSpec (',' RoleSpec)*  
  
IfNotExists ::=  
  ('IF' 'NOT' 'EXISTS')?  
  
RoleSpec ::=  
  Rolename
```

11.5.2.26.2 Examples

Create a new role for the analytics team, and a new user called jennifer:

```
$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 37
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible

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> CREATE ROLE analyticsteam;
Query OK, 0 rows affected (0.02 sec)

mysql> GRANT SELECT ON test.* TO analyticsteam;
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE USER jennifer;
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT analyticsteam TO jennifer;
Query OK, 0 rows affected (0.01 sec)
```

Note that by default jennifer needs to SET ROLE analyticsteam in order to be able to use the privileges associated with the role:

```
$ mysql -ujennifer
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 32
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible

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> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
2 rows in set (0.00 sec)

mysql> SHOW TABLES in test;
ERROR 1044 (42000): Access denied for user 'jennifer'@'%' to database 'test'
mysql> SET ROLE analyticsteam;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT Select ON test.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
3 rows in set (0.00 sec)

mysql> SHOW TABLES IN test;
+-----+
| Tables_in_test |
+-----+
| t1           |
+-----+
1 row in set (0.00 sec)
```

The statement `SET DEFAULT ROLE` can be used to associate a role to `jennifer` so that she will not have to execute the statement `SET ROLE` in order to assume the privileges associated with the role:

```
$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 34
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible
```

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> SET DEFAULT ROLE analyticsteam TO jennifer;
Query OK, 0 rows affected (0.02 sec)
```

```
$ mysql -ujennifer
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 35
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
    ↵ License 2.0) Community Edition, MySQL 5.7 compatible
```

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> SHOW GRANTS;
+-----+
| Grants for User          |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%'  |
| GRANT Select ON test.* TO 'jennifer'@'%'  |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%'  |
+-----+
3 rows in set (0.00 sec)
```

```
mysql> SHOW TABLES IN test;
+-----+
| Tables_in_test |
+-----+
| t1           |
```

```
+-----+
1 row in set (0.00 sec)
```

11.5.2.26.3 MySQL compatibility

This statement is understood to be fully compatible with roles, which are a feature of MySQL 8.0. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.26.4 See also

- [DROP ROLE](#)
- [GRANT <role>](#)
- [REVOKE <role>](#)
- [SET ROLE](#)
- [SET DEFAULT ROLE](#)
- [Role-Based Access Control](#)

11.5.2.27 CREATE SEQUENCE

The `CREATE SEQUENCE` statement creates sequence objects in TiDB. The sequence is a database object that is on a par with the table and the `View` object. The sequence is used to generate serialized IDs in a customized way.

11.5.2.27.1 Synopsis

```
CreateSequenceStmt ::=  
  'CREATE' 'SEQUENCE' IfNotExists TableName CreateSequenceOptionListOpt  
    ↪ CreateTableOptionListOpt  
  
IfNotExists ::=  
  ('IF' 'NOT' 'EXISTS')?  
  
TableName ::=  
  Identifier ('.' Identifier)?  
  
CreateSequenceOptionListOpt ::=  
  SequenceOption*  
  
SequenceOptionList ::=  
  SequenceOption  
  
SequenceOption ::=  
  ('INCREMENT' ('='? | 'BY' ) | 'START' ('='? | 'WITH' ) | ('MINVALUE'  
    ↪ | 'MAXVALUE' | 'CACHE' ) '='? ) SignedNum
```

```
| 'NOMINVALUE'  
| 'NO' ( 'MINVALUE' | 'MAXVALUE' | 'CACHE' | 'CYCLE' )  
| 'NOMAXVALUE'  
| 'NOCACHE'  
| 'CYCLE'  
| 'NOCYCLE'
```

11.5.2.27.2 Syntax

```
CREATE [TEMPORARY] SEQUENCE [IF NOT EXISTS] sequence_name  
[ INCREMENT [ BY | = ] increment ]  
[ MINVALUE [=] minvalue | NO MINVALUE | NOMINVALUE ]  
[ MAXVALUE [=] maxvalue | NO MAXVALUE | NOMAXVALUE ]  
[ START [ WITH | = ] start ]  
[ CACHE [=] cache | NOCACHE | NO CACHE]  
[ CYCLE | NOCYCLE | NO CYCLE]  
[table_options]
```

11.5.2.27.3 Parameters

Parameter	Default	Description
TEMPORARY	ANALyse TiDB → → cur- rently does not sup- port the TEMPORARY → op- tion and pro- vides only syn- tax com- pati- bil- ity for it.	

Parameter	Default	Description
INCREMENT	Specifies the in- cre- ment of a se- quence. Its posi- tive or neg- a- tive val- ues can con- trol the growth di- rec- tion of the se- quence.	

Parameter	Default	Description
MINVALUE	Specifies -9223372036854775807	
	↪ min-	
	↪ i-	
	↪ mum	
	↪ value	
	↪ of a	
	↪ se-	
	↪ quence.	
	When	
	INCREMENT	
	↪	
	↪ > 0,	
	↪ the	
	↪ de-	
	↪ fault	
	↪ value	
	↪ is 1.	
	When	
	INCREMENT	
	↪	
	↪ < 0,	
	↪ the	
	↪ de-	
	↪ fault	
	↪ value	
	↪ is	
	-9223372036854775807	
	↪ .	

Parameter	Default	Description
MAXVAL	9223372036854775806	
→	→	the
-1	or	max-
		i-
		num
		value
		of a
		se-
		quence.
		When
		INCREMENT
	→	
	> 0,	
	the	
	de-	
	fault	
	value	
	is	
	9223372036854775806	
	→ .	
		When
		INCREMENT
	→	
	< 0,	
	the	
	de-	
	fault	
	value	
	is	
	-1.	

Parameter	Default	Description
START MINVALUE	Specifies → → the or ini- MAXVALUE → value of a se- quence. When INCREMENT → → > 0, the de- fault value is MINVALUE → . When INCREMENT → → < 0, the de- fault value is MAXVALUE → .	
CACHE 1000	Specifies → → the lo- cal cache size of a se- quence in TiDB.	

Parameter	Description	Default
CYCLE NO	Specifies the number of times the sequence restarts from the minimum value (or maximum value) for the descending sequence).	
INCREMENT	When INCREMENT is greater than 0, the default fault value is MINVALUE.	
MINVALUE	When INCREMENT is less than 0, the default fault value is MAXVALUE.	
MAXVALUE		

Parameter	Default	Description
-----------	---------	-------------

11.5.2.27.4 SEQUENCE function

You can control a sequence through the following expression functions:

- **NEXTVAL** or **NEXT VALUE FOR**

Essentially, both are the `nextval()` function that gets the next valid value of a sequence object. The parameter of the `nextval()` function is the `identifier` of the sequence.

- **LASTVAL**

This function gets the last used value of this session. If the value does not exist, `NULL` is used. The parameter of this function is the `identifier` of the sequence.

- **SETVAL**

This function sets the progression of the current value for a sequence. The first parameter of this function is the `identifier` of the sequence; the second parameter is `num`.

Note:

In the implementation of a sequence in TiDB, the `SETVAL` function cannot change the initial progression or cycle progression of this sequence. This function only returns the next valid value based on this progression.

11.5.2.27.5 Examples

- Create a sequence object with the default parameter:

```
CREATE SEQUENCE seq;
```

```
Query OK, 0 rows affected (0.06 sec)
```

- Use the `nextval()` function to get the next value of the sequence object:

```
SELECT nextval(seq);
```

```
+-----+
| nextval(seq) |
+-----+
|      1      |
+-----+
1 row in set (0.02 sec)
```

- Use the `lastval()` function to get the value generated by the last call to a sequence object in this session:

```
SELECT lastval(seq);
```

```
+-----+
| lastval(seq) |
+-----+
|      1      |
+-----+
1 row in set (0.02 sec)
```

- Use the `setval()` function to set the current value (or the current position) of the sequence object:

```
SELECT setval(seq, 10);
```

```
+-----+
| setval(seq, 10) |
+-----+
|      10      |
+-----+
1 row in set (0.01 sec)
```

- You can also use the `next value for` syntax to get the next value of the sequence:

```
SELECT next value for seq;
```

```
+-----+
| next value for seq |
+-----+
|      11      |
+-----+
1 row in set (0.00 sec)
```

- Create a sequence object with default custom parameters:

```
CREATE SEQUENCE seq2 start 3 increment 2 minvalue 1 maxvalue 10 cache
→ 3;
```

```
Query OK, 0 rows affected (0.01 sec)
```

- When the sequence object has not been used in this session, the `lastval()` function returns a `NULL` value.

```
SELECT lastval(seq2);
```

```
+-----+
| lastval(seq2) |
+-----+
|      NULL   |
+-----+
1 row in set (0.01 sec)
```

- The first valid value of the `nextval()` function for the sequence object is the value of `START` parameter.

```
SELECT nextval(seq2);
```

```
+-----+
| nextval(seq2) |
+-----+
|      3      |
+-----+
1 row in set (0.00 sec)
```

- Although the `setval()` function can change the current value of the sequence object, it cannot change the arithmetic progression rule for the next value.

```
SELECT setval(seq2, 6);
```

```
+-----+
| setval(seq2, 6) |
+-----+
|      6      |
+-----+
1 row in set (0.00 sec)
```

- When you use `nextval()` to get the next value, the next value will follow the arithmetic progression rule defined by the sequence.

```
SELECT next value for seq2;
```

```
+-----+
| next value for seq2 |
+-----+
|          7 |
+-----+
1 row in set (0.00 sec)
```

- You can use the next value of the sequence as the default value for the column, as in the following example.

```
CREATE table t(a int default next value for seq2);
```

```
Query OK, 0 rows affected (0.02 sec)
```

- In the following example, the value is not specified, so the default value of seq2 is used.

```
INSERT into t values();
```

```
Query OK, 1 row affected (0.00 sec)
```

```
SELECT * from t;
```

```
+----+
| a   |
+----+
|    9 |
+----+
1 row in set (0.00 sec)
```

- In the following example, the value is not specified, so the default value of seq2 is used. But the next value of seq2 is not within the range defined in the above example (CREATE SEQUENCE seq2 start 3 increment 2 minvalue 1 maxvalue 10 ↵ cache 3;), so an error is returned.

```
INSERT into t values();
```

```
ERROR 4135 (HY000): Sequence 'test.seq2' has run out
```

11.5.2.27.6 MySQL compatibility

This statement is a TiDB extension. The implementation is modeled on sequences available in MariaDB.

Except for the SETVAL function, all other functions have the same *progressions* as MariaDB. Here “progression” means that the numbers in a sequence follow a certain arithmetic

progression rule defined by the sequence. Although you can use `SETVAL` to set the current value of a sequence, the subsequent values of the sequence still follow the original progression rule.

For example:

```
1, 3, 5, ...          // The sequence starts from 1 and increments by 2.
select setval(seq, 6) // Sets the current value of a sequence to 6.
7, 9, 11, ...         // Subsequent values still follow the progression rule.
```

In the CYCLE mode, the initial value of a sequence in the first round is the value of the `START` parameter, and the initial value in the subsequent rounds is the value of `MinValue` (`INCREMENT > 0`) or `MaxValue` (`INCREMENT < 0`).

11.5.2.27.7 See also

- [DROP SEQUENCE](#)
- [SHOW CREATE SEQUENCE](#)

11.5.2.28 CREATE TABLE LIKE

This statement copies the definition of an existing table, without copying any data.

11.5.2.28.1 Synopsis

```
CreateTableLikeStmt ::= 
    'CREATE' OptTemporary 'TABLE' IfNotExists TableName
        ↪ LikeTableWithOrWithoutParen OnCommitOpt

OptTemporary ::= 
    ('TEMPORARY' | ('GLOBAL' 'TEMPORARY'))?

LikeTableWithOrWithoutParen ::= 
    'LIKE' TableName
    | '(' 'LIKE' TableName ')'

OnCommitOpt ::= 
    ('ON' 'COMMIT' 'DELETE' 'ROWS')?
```

11.5.2.28.2 Examples

```
mysql> CREATE TABLE t1 (a INT NOT NULL);
Query OK, 0 rows affected (0.13 sec)

mysql> INSERT INTO t1 VALUES (1),(2),(3),(4),(5);
```

```

Query OK, 5 rows affected (0.02 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM t1;
+---+
| a |
+---+
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
+---+
5 rows in set (0.00 sec)

mysql> CREATE TABLE t2 LIKE t1;
Query OK, 0 rows affected (0.10 sec)

mysql> SELECT * FROM t2;
Empty set (0.00 sec)

```

11.5.2.28.3 Pre-split region

If the table to be copied is defined with the `PRE_SPLIT_REGIONS` attribute, the table created using the `CREATE TABLE LIKE` statement inherits this attribute, and the Region on the new table will be split. For details of `PRE_SPLIT_REGIONS`, see [CREATE TABLE Statement](#).

11.5.2.28.4 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.28.5 See also

- [CREATE TABLE](#)
- [SHOW CREATE TABLE](#)

11.5.2.29 CREATE TABLE

This statement creates a new table in the currently selected database. It behaves similarly to the `CREATE TABLE` statement in MySQL.

11.5.2.29.1 Synopsis

```

CreateTableStmt ::=

  'CREATE' OptTemporary 'TABLE' IfNotExists TableName (
    ↪ TableElementListOpt CreateTableOptionListOpt PartitionOpt
    ↪ DuplicateOpt AsOpt CreateTableSelectOpt |
    ↪ LikeTableWithOrWithoutParen ) OnCommitOpt

OptTemporary ::=

  ('TEMPORARY' | ('GLOBAL' 'TEMPORARY'))?

IfNotExists ::=

  ('IF' 'NOT' 'EXISTS')?

TableName ::=

  Identifier ('.' Identifier)?

TableElementListOpt ::=

  ('(' TableElementList ')')?

TableElementList ::=

  TableElement ( ',' TableElement )*

TableElement ::=

  ColumnDef
  | Constraint

ColumnDef ::=

  ColumnName ( Type | 'SERIAL' ) ColumnOptionListOpt

ColumnOptionListOpt ::=

  ColumnOption*

ColumnOptionList ::=

  ColumnOption*

ColumnOption ::=

  'NOT'? 'NULL'
  | 'AUTO_INCREMENT'
  | PrimaryOpt 'KEY'
  | 'UNIQUE' 'KEY'?
  | 'DEFAULT' DefaultValueExpr
  | 'SERIAL' 'DEFAULT' 'VALUE'
  | 'ON' 'UPDATE' NowSymOptionFraction
  | 'COMMENT' stringLit

```

```

| ConstraintKeywordOpt 'CHECK' '(' Expression ')'
|   ↪ EnforcedOrNotOrNotNullOpt
| GeneratedAlways 'AS' '(' Expression ')' VirtualOrStored
| ReferDef
| 'COLLATE' CollationName
| 'COLUMN_FORMAT' ColumnFormat
| 'STORAGE' StorageMedia
| 'AUTO_RANDOM' OptFieldLen

CreateTableOptionListOpt ::= TableOptionList?

PartitionOpt ::= ('PARTITION' 'BY' PartitionMethod PartitionNumOpt SubPartitionOpt
                  ↪ PartitionDefinitionListOpt )?

DuplicateOpt ::= ('IGNORE' | 'REPLACE')?

TableOptionList ::= TableOption ( ','? TableOption )*

TableOption ::= PartDefOption
| DefaultKwdOpt ( CharsetKw EqOpt CharsetName | 'COLLATE' EqOpt
                  ↪ CollationName )
| ('AUTO_INCREMENT' | 'AUTO_ID_CACHE' | 'AUTO_RANDOM_BASE' | '
                  ↪ AVG_ROW_LENGTH' | 'CHECKSUM' | 'TABLE_CHECKSUM' | 'KEY_BLOCK_SIZE' | '
                  ↪ 'DELAY_KEY_WRITE' | 'SHARD_ROW_ID_BITS' | 'PRE_SPLIT_REGIONS' ) EqOpt
                  ↪ LengthNum
| ('CONNECTION' | 'PASSWORD' | 'COMPRESSION' ) EqOpt stringLit
| RowFormat
| ('STATS_PERSISTENT' | 'PACK_KEYS' ) EqOpt StatsPersistentVal
| ('STATS_AUTO_RECALC' | 'STATS_SAMPLE_PAGES' ) EqOpt ( LengthNum | '
                  ↪ DEFAULT' )
| 'STORAGE' ( 'MEMORY' | 'DISK' )
| 'SECONDARY_ENGINE' EqOpt ( 'NULL' | StringName )
| 'UNION' EqOpt '(' TableNameListOpt ')'
| 'ENCRYPTION' EqOpt EncryptionOpt

OnCommitOpt ::= ('ON' 'COMMIT' 'DELETE' 'ROWS')?

```

The following *table_options* are supported. Other options such as AVG_ROW_LENGTH
 ↪ , CHECKSUM, COMPRESSION, CONNECTION, DELAY_KEY_WRITE, ENGINE, KEY_BLOCK_SIZE,

`MAX_ROWS`, `MIN_ROWS`, `ROW_FORMAT` and `STATS_PERSISTENT` are parsed but ignored.

Options	Description	Example
<code>AUTO_INCREMENT</code>	The initial value of the increment field	<code>AUTO_INCREMENT</code>
<code>SHARD_ROW_ID_BITS</code>	Specifies the number of bits for the implicit <code>_tidb_rowid</code> shards	<code>SHARD_ROW_ID_BITS</code>
<code>PRE_SPLIT_REGIONS</code>	Regions-split when creating a table	<code>PRE_SPLIT_REGIONS</code>
<code>AUTO_ID_CACHE</code>	To set the auto ID cache size in a TiDB instance. By default, TiDB automatically changes this size according to allocation speed of auto ID	<code>AUTO_ID_CACHE</code>

Options	Description	Example
AUTO_RANDOM_BASE	Set the initial incremental part value of auto_random. This option can be considered as a part of the internal interface. Users can ignore this parameter.	AUTO_RANDOM_BASE ↪ = 0
CHARACTER	To specify the character set for the table	CHARACTER ↪ SET ↪ character set = 'utf8mb4'
COMMENT	The comment information	COMMENT ↪ = 'comment info'

Note:

The `split-table` configuration option is enabled by default. When it is enabled, a separate Region is created for each newly created table. For details, see [TiDB configuration file](#).

11.5.2.29.2 Examples

Creating a simple table and inserting one row:

```
CREATE TABLE t1 (a int);
DESC t1;
SHOW CREATE TABLE t1\G
INSERT INTO t1 (a) VALUES (1);
SELECT * FROM t1;
```

```

mysql> drop table if exists t1;
Query OK, 0 rows affected (0.23 sec)

mysql> CREATE TABLE t1 (a int);
Query OK, 0 rows affected (0.09 sec)

mysql> DESC t1;
+-----+-----+-----+-----+
| Field | Type  | Null | Key | Default | Extra |
+-----+-----+-----+-----+
| a     | int(11) | YES  |      | NULL    |       |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> SHOW CREATE TABLE t1\G
***** 1. row *****
    Table: t1
Create Table: CREATE TABLE `t1` (
  `a` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin
1 row in set (0.00 sec)

mysql> INSERT INTO t1 (a) VALUES (1);
Query OK, 1 row affected (0.03 sec)

mysql> SELECT * FROM t1;
+---+
| a |
+---+
| 1 |
+---+
1 row in set (0.00 sec)

```

Dropping a table if it exists, and conditionally creating a table if it does not exist:

```

DROP TABLE IF EXISTS t1;
CREATE TABLE IF NOT EXISTS t1 (
  id BIGINT NOT NULL PRIMARY KEY auto_increment,
  b VARCHAR(200) NOT NULL
);
DESC t1;

```

```

mysql> DROP TABLE IF EXISTS t1;
Query OK, 0 rows affected (0.22 sec)

```

```

mysql> CREATE TABLE IF NOT EXISTS t1 (
    -> id BIGINT NOT NULL PRIMARY KEY auto_increment,
    -> b VARCHAR(200) NOT NULL
    -> );
Query OK, 0 rows affected (0.08 sec)

mysql> DESC t1;
+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra       |
+-----+-----+-----+-----+
| id   | bigint(20) | NO   | PRI | NULL    | auto_increment |
| b    | varchar(200)| NO  |     | NULL    |              |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)

```

11.5.2.29.3 MySQL compatibility

- All of the data types except spatial types are supported.
- FULLTEXT, HASH and SPATIAL indexes are not supported.
- For compatibility, the `index_col_name` attribute supports the length option with a maximum length limit of 3072 bytes by default. The length limit can be changed through the `max-index-length` configuration option. For details, see [TiDB configuration file](#).
- The `[ASC | DESC]` in `index_col_name` is currently parsed but ignored (MySQL 5.7 compatible behavior).
- The `COMMENT` attribute supports a maximum of 1024 characters and does not support the `WITH PARSER` option.
- TiDB supports at most 512 columns in a single table. The corresponding number limit in InnoDB is 1017, and the hard limit in MySQL is 4096. For details, see [TiDB Limitations](#).
- For partitioned tables, only Range, Hash and Range Columns (single column) are supported. For details, see [partitioned table](#).
- CHECK constraints are parsed but ignored (MySQL 5.7 compatible behavior). For details, see [Constraints](#).
- FOREIGN KEY constraints are parsed and stored, but not enforced by DML statements. For details, see [Constraints](#).

11.5.2.29.4 See also

- [Data Types](#)
- [DROP TABLE](#)
- [CREATE TABLE LIKE](#)
- [SHOW CREATE TABLE](#)

11.5.2.30 CREATE USER

This statement creates a new user, specified with a password. In the MySQL privilege system, a user is the combination of a username and the host from which they are connecting from. Thus, it is possible to create a user 'newuser2'@'192.168.1.1' who is only able to connect from the IP address 192.168.1.1. It is also possible to have two users have the same user-portion, and different permissions as they login from different hosts.

11.5.2.30.1 Synopsis

```

CreateUserStmt ::=

  'CREATE' 'USER' IfNotExists UserSpecList RequireClauseOpt
    ↪ ConnectionOptions PasswordOrLockOptions

IfNotExists ::=

  ('IF' 'NOT' 'EXISTS')?

UserSpecList ::=

  UserSpec ( ',' UserSpec )*

UserSpec ::=

  Username AuthOption

AuthOption ::=

  ( 'IDENTIFIED' ( 'BY' ( AuthString | 'PASSWORD' HashString ) | 'WITH'
    ↪ StringName ( 'BY' AuthString | 'AS' HashString )? )? )

StringName ::=

  stringLit
  | Identifier

```

11.5.2.30.2 Examples

Create a user with the newuserpassword password.

```
mysql> CREATE USER 'newuser' IDENTIFIED BY 'newuserpassword';
Query OK, 1 row affected (0.04 sec)
```

Create a user who can only log in to 192.168.1.1.

```
mysql> CREATE USER 'newuser2'@'192.168.1.1' IDENTIFIED BY 'newuserpassword'
  ↪ ;
Query OK, 1 row affected (0.02 sec)
```

Create a user who is enforced to log in using TLS connection.

```
CREATE USER 'newuser3'@'%' REQUIRE SSL IDENTIFIED BY 'newuserpassword';
Query OK, 1 row affected (0.02 sec)
```

Create a user who is required to use X.509 certificate at login.

```
CREATE USER 'newuser4'@'%' REQUIRE ISSUER '/C=US/ST=California/L=San
→ Francisco/O=PingCAP' IDENTIFIED BY 'newuserpassword';
Query OK, 1 row affected (0.02 sec)
```

11.5.2.30.3 MySQL compatibility

The following CREATE USER options are not yet supported by TiDB, and will be parsed but ignored:

- TiDB does not support WITH MAX_QUERIES_PER_HOUR, WITH MAX_UPDATES_PER_HOUR, and WITH MAX_USER_CONNECTIONS options.
- TiDB does not support the DEFAULT ROLE option.
- TiDB does not support PASSWORD EXPIRE, PASSWORD HISTORY or other options related to password.
- TiDB does not support the ACCOUNT LOCK and ACCOUNT UNLOCK options.

11.5.2.30.4 See also

- [Security Compatibility with MySQL](#)
- [DROP USER](#)
- [SHOW CREATE USER](#)
- [ALTER USER](#)
- [Privilege Management](#)

11.5.2.31 CREATE VIEW

The CREATE VIEW statement saves a SELECT statement as a queryable object, similar to a table. Views in TiDB are non-materialized. This means that as a view is queried, TiDB will internally rewrite the query to combine the view definition with the SQL query.

11.5.2.31.1 Synopsis

```
CreateViewStmt ::=

  'CREATE' OrReplace ViewAlgorithm ViewDefiner ViewSQLSecurity 'VIEW'
    → ViewName ViewFieldList 'AS' CreateViewSelectOpt ViewCheckOption

OrReplace ::=

  ( 'OR' 'REPLACE' )?
```

```

ViewAlgorithm ::= 
  ( 'ALGORITHM' '=' ( 'UNDEFINED' | 'MERGE' | 'TEMPTABLE' ) )?

ViewDefiner ::= 
  ( 'DEFINER' '=' Username )?

ViewSQLSecurity ::= 
  ( 'SQL' 'SECURITY' ( 'DEFINER' | 'INVOKER' ) )?

ViewName ::= TableName

ViewFieldList ::= 
  ( '(' Identifier ( ',' Identifier )* ')' )?

ViewCheckOption ::= 
  ( 'WITH' ( 'CASCADED' | 'LOCAL' ) 'CHECK' 'OPTION' )?

```

11.5.2.31.2 Examples

```

mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT
      ↪ NOT NULL);
Query OK, 0 rows affected (0.11 sec)

mysql> INSERT INTO t1 (c1) VALUES (1),(2),(3),(4),(5);
Query OK, 5 rows affected (0.03 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> CREATE VIEW v1 AS SELECT * FROM t1 WHERE c1 > 2;
Query OK, 0 rows affected (0.11 sec)

mysql> SELECT * FROM t1;
+----+----+
| id | c1 |
+----+----+
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
+----+----+
5 rows in set (0.00 sec)

mysql> SELECT * FROM v1;
+----+----+

```

```

| id | c1 |
+----+---+
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
+----+---+
3 rows in set (0.00 sec)

mysql> INSERT INTO t1 (c1) VALUES (6);
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM v1;
+----+---+
| id | c1 |
+----+---+
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
| 6 | 6 |
+----+---+
4 rows in set (0.00 sec)

mysql> INSERT INTO v1 (c1) VALUES (7);
ERROR 1105 (HY000): insert into view v1 is not supported now.

```

11.5.2.31.3 MySQL compatibility

- Currently, any view in TiDB cannot be inserted or updated (that is, `INSERT VIEW` and `UPDATE VIEW` are not supported). `WITH CHECK OPTION` is only syntactically compatible but does not take effect.
- Currently, the view in TiDB does not support `ALTER VIEW`, but you can use `CREATE → OR REPLACE` instead.
- Currently, the `ALGORITHM` field is only syntactically compatible in TiDB but does not take effect. TiDB currently only supports the `MERGE` algorithm.

11.5.2.31.4 See also

- [DROP VIEW](#)
- [CREATE TABLE](#)
- [SHOW CREATE TABLE](#)
- [DROP TABLE](#)

11.5.2.32 DEALLOCATE

The DEALLOCATE statement provides an SQL interface to server-side prepared statements.

11.5.2.32.1 Synopsis

```

DeallocateStmt ::=

    DeallocateSym 'PREPARE' Identifier

DeallocateSym ::=

    'DEALLOCATE'
  | 'DROP'

Identifier ::=

    identifier
  | UnReservedKeyword
  | NotKeywordToken
  | TiDBKeyword

```

11.5.2.32.2 Examples

```

mysql> PREPARE mystmt FROM 'SELECT ? as num FROM DUAL';
Query OK, 0 rows affected (0.00 sec)

mysql> SET @number = 5;
Query OK, 0 rows affected (0.00 sec)

mysql> EXECUTE mystmt USING @number;
+---+
| num |
+---+
| 5   |
+---+
1 row in set (0.00 sec)

mysql> DEALLOCATE PREPARE mystmt;
Query OK, 0 rows affected (0.00 sec)

```

11.5.2.32.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.32.4 See also

- [PREPARE](#)
- [EXECUTE](#)

11.5.2.33 DELETE

The DELETE statement removes rows from a specified table.

11.5.2.33.1 Synopsis

```
DeleteFromStmt ::=

  'DELETE' TableOptimizerHints PriorityOpt QuickOptional IgnoreOptional (
    ↪ 'FROM' ( TableName TableAsNameOpt IndexHintListOpt
    ↪ WhereClauseOptional OrderByOptional LimitClause |
    ↪ TableAliasRefList 'USING' TableRefs WhereClauseOptional ) |
    ↪ TableAliasRefList 'FROM' TableRefs WhereClauseOptional )
```

11.5.2.33.2 Examples

```
mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT
   ↪ NOT NULL);
Query OK, 0 rows affected (0.11 sec)

mysql> INSERT INTO t1 (c1) VALUES (1),(2),(3),(4),(5);
Query OK, 5 rows affected (0.03 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM t1;
+----+---+
| id | c1 |
+----+---+
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
+----+---+
5 rows in set (0.00 sec)

mysql> DELETE FROM t1 WHERE id = 4;
Query OK, 1 row affected (0.02 sec)

mysql> SELECT * FROM t1;
+----+---+
```

```

| id | c1 |
+----+---+
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 5 | 5 |
+----+---+
4 rows in set (0.00 sec)

```

11.5.2.33.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.33.4 See also

- [INSERT](#)
- [SELECT](#)
- [UPDATE](#)
- [REPLACE](#)

11.5.2.34 DESC

This statement is an alias to [EXPLAIN](#). It is included for compatibility with MySQL.

11.5.2.35 DESCRIBE

This statement is an alias to [EXPLAIN](#). It is included for compatibility with MySQL.

11.5.2.36 DO

`DO` executes the expressions but does not return any results. In most cases, `DO` is equivalent to `SELECT expr, ...` that does not return a result.

Note:

`DO` only executes expressions. It cannot be used in all cases where `SELECT` can be used. For example, `DO id FROM t1` is invalid because it references a table.

In MySQL, a common use case is to execute stored procedure or trigger. Since TiDB does not provide stored procedure or trigger, this function has a limited use.

11.5.2.36.1 Synopsis

```

DoStmt ::= 'DO' ExpressionList

ExpressionList :=
    Expression ( ',' Expression )*

Expression :=
    ( singleAtIdentifier assignmentEq | 'NOT' | Expression ( logOr | 'XOR' |
        ↪ logAnd ) ) Expression
| 'MATCH' '(' ColumnNameList ')' 'AGAINST' '(' BitExpr
    ↪ FulltextSearchModifierOpt ')'
| PredicateExpr ( IsOrNotOp 'NULL' | CompareOp ( ( singleAtIdentifier
    ↪ assignmentEq )? PredicateExpr | AnyOrAll SubSelect ) )* ( IsOrNotOp (
    ↪ trueKwd | falseKwd | 'UNKNOWN' ) )?

```

11.5.2.36.2 Examples

This SELECT statement pauses, but also produces a result set.

```

mysql> SELECT SLEEP(5);
+-----+
| SLEEP(5) |
+-----+
|      0   |
+-----+
1 row in set (5.00 sec)

```

DO, on the other hand, pauses without producing a result set.

```

mysql> DO SLEEP(5);
Query OK, 0 rows affected (5.00 sec)

mysql> DO SLEEP(1), SLEEP(1.5);
Query OK, 0 rows affected (2.50 sec)

```

11.5.2.36.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be [reported via an issue](#) on GitHub.

11.5.2.36.4 See also

- [SELECT](#)

11.5.2.37 DROP [GLOBAL|SESSION] BINDING

This statement removes a binding from a specific SQL statement. Bindings can be used to inject a hint into a statement without requiring changes to the underlying query.

A BINDING can be on either a GLOBAL or SESSION basis. The default is SESSION.

11.5.2.37.1 Synopsis

```

DropBindingStmt ::=

    'DROP' GlobalScope 'BINDING' 'FOR' BindableStmt ( 'USING' BindableStmt )
    ↪ ?

GlobalScope ::=

    ( 'GLOBAL' | 'SESSION' )?

BindableStmt ::=

    ( SelectStmt | UpdateStmt | InsertIntoStmt | ReplaceIntoStmt |
      ↪ DeleteStmt )

```

11.5.2.37.2 Syntax description

```

mysql> CREATE TABLE t1 (
    -> id INT NOT NULL PRIMARY KEY auto_increment,
    -> b INT NOT NULL,
    -> pad VARBINARY(255),
    -> INDEX(b)
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    ↪ FROM dual;
Query OK, 1 row affected (0.01 sec)
Records: 1 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 1 row affected (0.00 sec)
Records: 1 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 8 rows affected (0.00 sec)
Records: 8 Duplicates: 0 Warnings: 0

```

```

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
   ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 1000 rows affected (0.04 sec)
Records: 1000 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
   ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 100000 rows affected (1.74 sec)
Records: 100000 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
   ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 100000 rows affected (2.15 sec)
Records: 100000 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
   ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 100000 rows affected (2.64 sec)
Records: 100000 Duplicates: 0 Warnings: 0

mysql> SELECT SLEEP(1);
+-----+
| SLEEP(1) |
+-----+
|      0   |
+-----+
1 row in set (1.00 sec)

mysql> ANALYZE TABLE t1;
Query OK, 0 rows affected (1.33 sec)

mysql> EXPLAIN ANALYZE SELECT * FROM t1 WHERE b = 123;
+-----
→ -----+-----+-----+-----+-----+
→
| id           | estRows | actRows | task    | access object
→       | execution info
→
→               | memory     | disk    |          | operator info
→
+-----+
→ -----+-----+-----+-----+
→
| IndexLookUp_10        | 583.00 | 297    | root    |
→                   | time:10.545072ms, loops:2, rpc num: 1, rpc time
→ :398.359μs, proc keys:297 |          | 109.1484375 KB | N/
+-----+

```

```

    ↗ A |
| -IndexRangeScan_8(Build) | 583.00 | 297 | cop[tikv] | table:t1, index
    ↗ :b(b) | time:0s, loops:4
    ↗ range:[123,123], keep order:false | N/A | N/A |
| -TableRowIDScan_9(Probe) | 583.00 | 297 | cop[tikv] | table:t1
    ↗ | time:12ms, loops:4
    ↗ | keep order:false
    ↗ | N/A | N/A |
+--+
    ↗ -----
    ↗
3 rows in set (0.02 sec)

mysql> CREATE SESSION BINDING FOR
-> SELECT * FROM t1 WHERE b = 123
-> USING
-> SELECT * FROM t1 IGNORE INDEX (b) WHERE b = 123;
Query OK, 0 rows affected (0.00 sec)

mysql> EXPLAIN ANALYZE SELECT * FROM t1 WHERE b = 123;
+--+
    ↗ -----
    ↗
| id           | estRows | actRows | task      | access object |
| execution info
| operator info | memory   | disk    |
+--+
    ↗ -----
    ↗
| TableReader_7 | 583.00 | 297   | root     |               | time
    ↗ :222.32506ms, loops:2, rpc num: 1, rpc time:222.078952ms, proc keys
    ↗ :301010 | data:Selection_6 | 88.6640625 KB | N/A |
| -Selection_6  | 583.00 | 297   | cop[tikv] |               | time
    ↗ :224ms, loops:298
    ↗ test.t1.b, 123) | N/A     | N/A |
| -TableFullScan_5 | 301010.00 | 301010 | cop[tikv] | table:t1 | time
    ↗ :220ms, loops:298
    ↗ keep order:false | N/A     | N/A |
+--+
    ↗ -----
    ↗
3 rows in set (0.22 sec)

mysql> SHOW SESSION BINDINGS\G
***** 1. row *****

```

```

Original_sql: select * from t1 where b = ?
Bind_sql: SELECT * FROM t1 IGNORE INDEX (b) WHERE b = 123
Default_db: test
Status: using
Create_time: 2020-05-22 14:38:03.456
Update_time: 2020-05-22 14:38:03.456
Charset: utf8mb4
Collation: utf8mb4_0900_ai_ci
1 row in set (0.00 sec)

mysql> DROP SESSION BINDING FOR SELECT * FROM t1 WHERE b = 123;
Query OK, 0 rows affected (0.00 sec)

mysql> EXPLAIN ANALYZE SELECT * FROM t1 WHERE b = 123;
+---+
→ | id           | estRows | actRows | task   | access object
→ |             | execution info
→ | operator info          | memory    | disk   |
+---+
→ | IndexLookUp_10      | 583.00  | 297    | root   |
→ |                      | time:5.31206ms, loops:2, rpc num: 1, rpc time
→ | :665.927µs, proc keys:297 |          | 109.1484375 KB | N
→ | /A |
| -IndexRangeScan_8(Build) | 583.00  | 297    | cop[tikv] | table:t1, index
→ | :b(b) | time:0s, loops:4
→ | range:[123,123], keep order:false | N/A     | N/A   |
| -TableRowIDScan_9(Probe) | 583.00  | 297    | cop[tikv] | table:t1
→ |          | time:0s, loops:4
→ |          | N/A       | N/A   |
+---+
→ | N/A       | N/A   |
3 rows in set (0.01 sec)

mysql> SHOW SESSION BINDINGS\G
Empty set (0.00 sec)

```

11.5.2.37.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.37.4 See also

- [CREATE \[GLOBAL|SESSION\] BINDING](#)
- [SHOW \[GLOBAL|SESSION\] BINDINGS](#)
- [ANALYZE TABLE](#)
- [Optimizer Hints](#)
- [SQL Plan Management](#)

11.5.2.38 DROP COLUMN

This statement drops a column from a specified table. `DROP COLUMN` is online in TiDB, which means that it does not block read or write operations.

11.5.2.38.1 Synopsis

```

AlterTableStmt ::=

  'ALTER' IgnoreOptional 'TABLE' TableName ( AlterTableSpecListOpt
    ↪ AlterTablePartitionOpt | 'ANALYZE' 'PARTITION' PartitionNameList (
    ↪ 'INDEX' IndexNameList )? AnalyzeOptionListOpt )

AlterTableSpec ::=

  TableOptionList

  | 'SET' 'TIFLASH' 'REPLICA' LengthNum LocationLabelList
  | 'CONVERT' 'TO' CharsetKw ( CharsetName | 'DEFAULT' ) OptCollate
  | 'ADD' ( ColumnKeywordOpt IfNotExists ( ColumnDef ColumnPosition | '('
    ↪ TableElementList ')' ) | Constraint | 'PARTITION' IfNotExists
    ↪ NoWriteToBinLogAliasOpt ( PartitionDefinitionListOpt | 'PARTITIONS'
    ↪ NUM ) )
  | ( ( 'CHECK' | 'TRUNCATE' ) 'PARTITION' | ( 'OPTIMIZE' | 'REPAIR' | '
    ↪ REBUILD' ) 'PARTITION' NoWriteToBinLogAliasOpt )
    ↪ AllOrPartitionNameList
  | 'COALESCE' 'PARTITION' NoWriteToBinLogAliasOpt NUM
  | 'DROP' ( ColumnKeywordOpt IfExists ColumnName RestrictOrCascadeOpt | '
    ↪ PRIMARY' 'KEY' | 'PARTITION' IfExists PartitionNameList | (
    ↪ KeyOrIndex IfExists | 'CHECK' ) Identifier | 'FOREIGN' 'KEY' IfExists
    ↪ Symbol )
  | 'EXCHANGE' 'PARTITION' Identifier 'WITH' 'TABLE' TableName
    ↪ WithValidationOpt
  | ( 'IMPORT' | 'DISCARD' ) ( 'PARTITION' AllOrPartitionNameList )? '
    ↪ TABLESPACE'
  | 'REORGANIZE' 'PARTITION' NoWriteToBinLogAliasOpt
    ↪ ReorganizePartitionRuleOpt
  | 'ORDER' 'BY' AlterOrderItem ( ',' AlterOrderItem )*

```

```

| ( 'DISABLE' | 'ENABLE' ) 'KEYS'
| ( 'MODIFY' ColumnKeywordOpt IfExists | 'CHANGE' ColumnKeywordOpt
  ↪ IfExists ColumnName ) ColumnDef ColumnPosition
| 'ALTER' ( ColumnKeywordOpt ColumnName ( 'SET' 'DEFAULT' ( SignedLiteral
  ↪ | '(' Expression ')' ) | 'DROP' 'DEFAULT' ) | 'CHECK' Identifier
  ↪ EnforcedOrNot | 'INDEX' Identifier IndexInvisible )
| 'RENAME' ( ( 'COLUMN' | KeyOrIndex ) Identifier 'TO' Identifier | ( 'TO'
  ↪ | '='? | 'AS' ) TableName )
| LockClause
| AlgorithmClause
| 'FORCE'
| ( 'WITH' | 'WITHOUT' ) 'VALIDATION'
| 'SECONDARY_LOAD'
| 'SECONDARY_UNLOAD'

ColumnName ::==
  Identifier ( '.' Identifier ( '.' Identifier )? )?

```

11.5.2.38.2 Examples

```

mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, col1
   ↪ INT NOT NULL, col2 INT NOT NULL);
Query OK, 0 rows affected (0.12 sec)

mysql> INSERT INTO t1 (col1,col2) VALUES (1,1),(2,2),(3,3),(4,4),(5,5);
Query OK, 5 rows affected (0.02 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM t1;
+----+----+
| id | col1 | col2 |
+----+----+
| 1 | 1 | 1 |
| 2 | 2 | 2 |
| 3 | 3 | 3 |
| 4 | 4 | 4 |
| 5 | 5 | 5 |
+----+----+
5 rows in set (0.01 sec)

mysql> ALTER TABLE t1 DROP COLUMN col1, DROP COLUMN col2;
ERROR 1105 (HY000): can't run multi schema change
mysql> SELECT * FROM t1;
+----+----+

```

```

| id | col1 | col2 |
+---+---+---+
| 1 | 1 | 1 |
| 2 | 2 | 2 |
| 3 | 3 | 3 |
| 4 | 4 | 4 |
| 5 | 5 | 5 |
+---+---+---+
5 rows in set (0.00 sec)

mysql> ALTER TABLE t1 DROP COLUMN col1;
Query OK, 0 rows affected (0.27 sec)

mysql> SELECT * FROM t1;
+---+---+
| id | col2 |
+---+---+
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
+---+---+
5 rows in set (0.00 sec)

```

11.5.2.38.3 MySQL compatibility

- Dropping multiple columns in the same statement is not supported.
- Dropping primary key columns or columns covered by the composite index is not supported.

11.5.2.38.4 See also

- [ADD COLUMN](#)
- [SHOW CREATE TABLE](#)
- [CREATE TABLE](#)

11.5.2.39 DROP DATABASE

The `DROP DATABASE` statement permanently removes a specified database schema, and all of the tables and views that were created inside. User privileges that are associated with the dropped database remain unaffected.

11.5.2.39.1 Synopsis

```
DropDatabaseStmt ::=  
    'DROP' 'DATABASE' IfExists DBName  
  
IfExists ::= ( 'IF' 'EXISTS' )?
```

11.5.2.39.2 Examples

```
mysql> SHOW DATABASES;  
+-----+  
| Database      |  
+-----+  
| INFORMATION_SCHEMA |  
| PERFORMANCE_SCHEMA |  
| mysql          |  
| test           |  
+-----+  
4 rows in set (0.00 sec)  
  
mysql> DROP DATABASE test;  
Query OK, 0 rows affected (0.25 sec)  
  
mysql> SHOW DATABASES;  
+-----+  
| Database      |  
+-----+  
| INFORMATION_SCHEMA |  
| PERFORMANCE_SCHEMA |  
| mysql          |  
+-----+  
3 rows in set (0.00 sec)
```

11.5.2.39.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be [reported via an issue](#) on GitHub.

11.5.2.39.4 See also

- [CREATE DATABASE](#)
- [ALTER DATABASE](#)

11.5.2.40 DROP INDEX

This statement removes an index from a specified table, marking space as free in TiKV.

11.5.2.40.1 Synopsis

```

AlterTableDropIndexStmt ::=

    'ALTER' IgnoreOptional 'TABLE' AlterTableDropIndexSpec

IgnoreOptional ::=

    'IGNORE'?

TableName ::=

    Identifier ('.' Identifier)?

AlterTableDropIndexSpec ::=

    'DROP' ( KeyOrIndex | 'FOREIGN' 'KEY' ) IfExists Identifier

KeyOrIndex ::=

    'KEY'
    |
    'INDEX'

IfExists ::= ( 'IF' 'EXISTS' )?

```

11.5.2.40.2 Examples

```

mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT
      ↪ NOT NULL);
Query OK, 0 rows affected (0.10 sec)

mysql> INSERT INTO t1 (c1) VALUES (1),(2),(3),(4),(5);
Query OK, 5 rows affected (0.02 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> EXPLAIN SELECT * FROM t1 WHERE c1 = 3;
+--+
| id          | estRows | task      | access object | operator info
|             |          |           |              |               |
+--+
| TableReader_7 | 10.00   | root      |              | data:Selection_6
|             |          |           |              |
+--+

```

```

|   -Selection_6      | 10.00 | cop[tikv] | eq(test.t1.c1,
|     ↵ 3)           |
|   -TableFullScan_5 | 10000.00 | cop[tikv] | table:t1 | keep order:false
|     ↵ , stats:pseudo |
+--+
|     ↵ -----+-----+-----+-----+
|     ↵
3 rows in set (0.00 sec)

mysql> CREATE INDEX c1 ON t1 (c1);
Query OK, 0 rows affected (0.30 sec)

mysql> EXPLAIN SELECT * FROM t1 WHERE c1 = 3;
+--+
| id          | estRows | task      | access object      | operator
| info        |          |           |                   |
+--+
|     ↵ -----+-----+-----+-----+
|     ↵
| IndexReader_6 | 0.01   | root      |                   | index:
|     ↵ IndexRangeScan_5
|   -IndexRangeScan_5 | 0.01   | cop[tikv] | table:t1, index:c1(c1) | range
|     ↵ :[3,3], keep order:false, stats:pseudo |
+--+
|     ↵ -----+-----+-----+-----+
|     ↵
2 rows in set (0.00 sec)

mysql> ALTER TABLE t1 DROP INDEX c1;
Query OK, 0 rows affected (0.30 sec)

```

11.5.2.40.3 MySQL compatibility

- Dropping the primary key of the CLUSTERED type is not supported. For more details about the primary key of the CLUSTERED type, refer to [clustered index](#).

11.5.2.40.4 See also

- [SHOW INDEX](#)
- [CREATE INDEX](#)
- [ADD INDEX](#)

- RENAME INDEX
- ALTER INDEX

11.5.2.41 DROP PLACEMENT POLICY

Warning:

Placement Rules in SQL is an experimental feature. The syntax might change before its GA, and there might also be bugs.

If you understand the risks, you can enable this experiment feature by executing `SET GLOBAL tidb_enable_alter_placement = 1;`.

DROP PLACEMENT POLICY is used to drop a previously created placement policy.

11.5.2.41.1 Synopsis

```
DropPolicyStmt ::=  
    "DROP" "PLACEMENT" "POLICY" IfExists PolicyName  
  
PolicyName ::=  
    Identifier
```

11.5.2.41.2 Examples

Placement policies can only be dropped when they are not referenced by any tables or partitions.

```
CREATE PLACEMENT POLICY p1 FOLLOWERS=4;  
CREATE TABLE t1 (a INT PRIMARY KEY) PLACEMENT POLICY=p1;  
DROP PLACEMENT POLICY p1; -- This statement fails because the placement  
    ↪ policy p1 is referenced.  
  
-- Finds which tables and partitions reference the placement policy.  
SELECT table_schema, table_name FROM information_schema.tables WHERE  
    ↪ tidb_placement_policy_name='p1';  
SELECT table_schema, table_name FROM information_schema.partitions WHERE  
    ↪ tidb_placement_policy_name='p1';  
  
ALTER TABLE t1 PLACEMENT POLICY=default; -- Removes the placement policy  
    ↪ from t1.  
DROP PLACEMENT POLICY p1; -- Succeeds.
```

```

Query OK, 0 rows affected (0.10 sec)

Query OK, 0 rows affected (0.11 sec)

ERROR 8241 (HY000): Placement policy 'p1' is still in use

+-----+
| table_schema | table_name |
+-----+
| test         | t1          |
+-----+
1 row in set (0.00 sec)

Empty set (0.01 sec)

Query OK, 0 rows affected (0.08 sec)

Query OK, 0 rows affected (0.21 sec)

```

11.5.2.41.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.41.4 See also

- [Placement Rules in SQL](#)
- [SHOW PLACEMENT](#)
- [CREATE PLACEMENT POLICY](#)
- [ALTER PLACEMENT POLICY](#)

11.5.2.42 DROP ROLE

This statement removes a role, that was previously created with CREATE ROLE.

11.5.2.42.1 Synopsis

```

DropRoleStmt ::=

  'DROP' 'ROLE' ( 'IF' 'EXISTS' )? RolenameList

RolenameList ::=

  Rolename ( ',' Rolename )*

```

11.5.2.42.2 Examples

Create a new role for the analytics team, and a new user called jennifer:

```
$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 37
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
  ↪ License 2.0) Community Edition, MySQL 5.7 compatible

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> CREATE ROLE analyticsteam;
Query OK, 0 rows affected (0.02 sec)

mysql> GRANT SELECT ON test.* TO analyticsteam;
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE USER jennifer;
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT analyticsteam TO jennifer;
Query OK, 0 rows affected (0.01 sec)
```

Note that by default jennifer needs to SET ROLE analyticsteam in order to be able to use the privileges associated with the role:

```
$ mysql -ujennifer
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 32
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
  ↪ License 2.0) Community Edition, MySQL 5.7 compatible

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> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
2 rows in set (0.00 sec)

mysql> SHOW TABLES in test;
ERROR 1044 (42000): Access denied for user 'jennifer'@'%' to database 'test'
mysql> SET ROLE analyticsteam;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT Select ON test.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
3 rows in set (0.00 sec)

mysql> SHOW TABLES IN test;
+-----+
| Tables_in_test |
+-----+
| t1           |
+-----+
1 row in set (0.00 sec)
```

The statement `SET DEFAULT ROLE` can be used to associate a role to `jennifer` so that she will not have to execute the statement `SET ROLE` in order to assume the privileges associated with the role:

```
$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 34
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible
```

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> SET DEFAULT ROLE analyticsteam TO jennifer;
Query OK, 0 rows affected (0.02 sec)
```

```
$ mysql -ujennifer
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 35
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
    ↵ License 2.0) Community Edition, MySQL 5.7 compatible
```

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> SHOW GRANTS;
+-----+
| Grants for User          |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT Select ON test.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
3 rows in set (0.00 sec)
```

```
mysql> SHOW TABLES IN test;
+-----+
| Tables_in_test |
+-----+
| t1           |
```

```
+-----+
1 row in set (0.00 sec)
```

Drop the role for the analyticsteam:

```
$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 41
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
    ↪ License 2.0) Community Edition, MySQL 5.7 compatible

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> DROP ROLE analyticsteam;
Query OK, 0 rows affected (0.02 sec)
```

Jennifer no longer has the default role of analyticsteam associated, or can set the role to analyticsteam:

```
$ mysql -ujennifer
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 42
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
    ↪ License 2.0) Community Edition, MySQL 5.7 compatible

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> SHOW GRANTS;
+-----+
| Grants for User |
```

```
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
+-----+
1 row in set (0.00 sec)

mysql> SET ROLE analyticsteam;
ERROR 3530 (HY000): `analyticsteam`@`%` is not granted to jennifer@%
```

11.5.2.42.3 MySQL compatibility

This statement is understood to be fully compatible with roles, which are a feature of MySQL 8.0. Any compatibility differences should be [reported via an issue](#) on GitHub.

11.5.2.42.4 See also

- [CREATE ROLE](#)
- [GRANT <role>](#)
- [REVOKE <role>](#)
- [SET ROLE](#)
- [SET DEFAULT ROLE](#)
- [Role-Based Access Control](#)

11.5.2.43 DROP SEQUENCE

The `DROP SEQUENCE` statement drops the sequence object in TiDB.

11.5.2.43.1 Synopsis

```
DropSequenceStmt ::=

  'DROP' 'SEQUENCE' IfExists TableNameList

IfExists ::= ( 'IF' 'EXISTS' )?

TableNameList ::=

  TableName ( ',' TableName )*

TableName ::=

  Identifier ('.' Identifier)?
```

11.5.2.43.2 Examples

```
DROP SEQUENCE seq;
```

```
Query OK, 0 rows affected (0.10 sec)
```

```
DROP SEQUENCE seq, seq2;
```

```
Query OK, 0 rows affected (0.03 sec)
```

11.5.2.43.3 MySQL compatibility

This statement is a TiDB extension. The implementation is modeled on sequences available in MariaDB.

11.5.2.43.4 See also

- [CREATE SEQUENCE](#)
- [SHOW CREATE SEQUENCE](#)

11.5.2.44 DROP STATS

The `DROP STATS` statement is used to delete the statistics of the selected table from the selected database.

11.5.2.44.1 Synopsis

```
DropStatsStmt ::=  
    'DROP' 'STATS' TableName
```

```
TableName ::=  
    Identifier ('.' Identifier)?
```

11.5.2.44.2 Examples

```
CREATE TABLE t(a INT);
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
SHOW STATS_META WHERE db_name='test' and table_name='t';
```

```
+--
```

```
→ -----+-----+-----+-----+
```

```
→
```

Db_name	Table_name	Partition_name	Update_time	Modify_count	Row_count
---------	------------	----------------	-------------	--------------	-----------

```
+--+
| test | t      |          | 2020-05-25 20:34:33 |      0 |
+--+
|      0 |
+--+
1 row in set (0.00 sec)
```

```
DROP STATS t;
```

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

```
SHOW STATS_META WHERE db_name='test' and table_name='t';
```

```
Empty set (0.00 sec)
```

11.5.2.44.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.44.4 See also

- [Introduction to Statistics](#)

11.5.2.45 DROP TABLE

This statement drops a table from the currently selected database. An error is returned if the table does not exist, unless the IF EXISTS modifier is used.

11.5.2.45.1 Synopsis

```
DropTableStmt ::=  
  'DROP' OptTemporary TableOrTables IfExists TableNameList  
    ↪ RestrictOrCascadeOpt  
  
OptTemporary ::=  
  ('TEMPORARY' | ('GLOBAL' 'TEMPORARY'))?  
  
TableOrTables ::=  
  'TABLE'  
|  'TABLES'
```

```
TableNameList ::=  
    TableName ( ',' TableName )*
```

11.5.2.45.2 Drop temporary tables

You can use the following syntax to drop ordinary tables and temporary tables:

- Use `DROP TEMPORARY TABLE` to drop local temporary tables.
- Use `DROP GLOBAL TEMPORARY TABLE` to drop global temporary tables.
- Use `DROP TABLE` to drop ordinary tables or temporary tables.

11.5.2.45.3 Examples

```
mysql> CREATE TABLE t1 (a INT);  
Query OK, 0 rows affected (0.11 sec)  
  
mysql> DROP TABLE t1;  
Query OK, 0 rows affected (0.22 sec)  
  
mysql> DROP TABLE table_not_exists;  
ERROR 1051 (42S02): Unknown table 'test.table_not_exists'  
mysql> DROP TABLE IF EXISTS table_not_exists;  
Query OK, 0 rows affected (0.01 sec)  
  
mysql> CREATE VIEW v1 AS SELECT 1;  
Query OK, 0 rows affected (0.10 sec)  
  
mysql> DROP TABLE v1;  
Query OK, 0 rows affected (0.23 sec)
```

11.5.2.45.4 MySQL compatibility

- Dropping a table with `IF EXISTS` does not return a warning when attempting to drop a table that does not exist. [Issue #7867](#)
- Currently `RESTRICT` and `CASCADE` are only supported syntactically.

11.5.2.45.5 See also

- [CREATE TABLE](#)
- [SHOW CREATE TABLE](#)
- [SHOW TABLES](#)

11.5.2.46 DROP USER

This statement removes a user from the TiDB system database. The optional keyword IF EXISTS can be used to silence an error if the user does not exist. This statement requires the CREATE USER privilege.

11.5.2.46.1 Synopsis

```
DropUserStmt ::=  
    'DROP' 'USER' ( 'IF' 'EXISTS' )? UsernameList  
  
Username ::=  
    StringName ('@' StringName | singleAtIdentifier)? | 'CURRENT_USER'  
        ↪ OptionalBraces
```

11.5.2.46.2 Examples

```
mysql> DROP USER idontexist;  
ERROR 1396 (HY000): Operation DROP USER failed for idontexist@%  
  
mysql> DROP USER IF EXISTS 'idontexist';  
Query OK, 0 rows affected (0.01 sec)  
  
mysql> CREATE USER 'newuser' IDENTIFIED BY 'mypassword';  
Query OK, 1 row affected (0.02 sec)  
  
mysql> GRANT ALL ON test.* TO 'newuser';  
Query OK, 0 rows affected (0.03 sec)  
  
mysql> SHOW GRANTS FOR 'newuser';  
+-----+  
| Grants for newuser@% |  
+-----+  
| GRANT USAGE ON *.* TO 'newuser'@'%' |  
| GRANT ALL PRIVILEGES ON test.* TO 'newuser'@'%' |  
+-----+  
2 rows in set (0.00 sec)  
  
mysql> REVOKE ALL ON test.* FROM 'newuser';  
Query OK, 0 rows affected (0.03 sec)  
  
mysql> SHOW GRANTS FOR 'newuser';  
+-----+  
| Grants for newuser@% |  
+-----+
```

```
| GRANT USAGE ON *.* TO 'newuser'@'%' |
+-----+
1 row in set (0.00 sec)

mysql> DROP USER 'newuser';
Query OK, 0 rows affected (0.14 sec)

mysql> SHOW GRANTS FOR 'newuser';
ERROR 1141 (42000): There is no such grant defined for user 'newuser' on
    ↪ host '%'
```

11.5.2.46.3 MySQL compatibility

- Dropping a user that does not exist with IF EXISTS will not create a warning in TiDB.
Issue #10196.

11.5.2.46.4 See also

- [CREATE USER](#)
- [ALTER USER](#)
- [SHOW CREATE USER](#)
- [Privilege Management](#)

11.5.2.47 DROP VIEW

This statement drops an view object from the currently selected database. It does not effect any base tables that a view references.

11.5.2.47.1 Synopsis

```
DropViewStmt ::=

  'DROP' 'VIEW' ( 'IF' 'EXISTS' )? TableNameList RestrictOrCascadeOpt

TableNameList ::=

  TableName ( ',' TableName )*

TableName ::=

  Identifier ('.' Identifier)?
```

11.5.2.47.2 Examples

```

mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT
      ↪ NOT NULL);
Query OK, 0 rows affected (0.11 sec)

mysql> INSERT INTO t1 (c1) VALUES (1),(2),(3),(4),(5);
Query OK, 5 rows affected (0.03 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> CREATE VIEW v1 AS SELECT * FROM t1 WHERE c1 > 2;
Query OK, 0 rows affected (0.11 sec)

mysql> SELECT * FROM t1;
+----+----+
| id | c1 |
+----+----+
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
+----+----+
5 rows in set (0.00 sec)

mysql> SELECT * FROM v1;
+----+----+
| id | c1 |
+----+----+
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
+----+----+
3 rows in set (0.00 sec)

mysql> DROP VIEW v1;
Query OK, 0 rows affected (0.23 sec)

mysql> SELECT * FROM t1;
+----+----+
| id | c1 |
+----+----+
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |

```

```

| 4 | 4 |
| 5 | 5 |
+----+----+
5 rows in set (0.00 sec)

```

11.5.2.47.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.47.4 See also

- [CREATE VIEW](#)
- [DROP TABLE](#)

11.5.2.48 EXECUTE

The EXECUTE statement provides an SQL interface to server-side prepared statements.

11.5.2.48.1 Synopsis

```
ExecuteStmt ::=  
  'EXECUTE' Identifier ( 'USING' UserVariable ( ',' UserVariable )* )?
```

11.5.2.48.2 Examples

```
mysql> PREPARE mystmt FROM 'SELECT ? as num FROM DUAL';
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> SET @number = 5;
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> EXECUTE mystmt USING @number;
+----+
| num |
+----+
| 5   |
+----+
1 row in set (0.00 sec)
```

```
mysql> DEALLOCATE PREPARE mystmt;
Query OK, 0 rows affected (0.00 sec)
```

11.5.2.48.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be [reported via an issue](#) on GitHub.

11.5.2.48.4 See also

- [PREPARE](#)
- [DEALLOCATE](#)

11.5.2.49 EXPLAIN ANALYZE

The EXPLAIN ANALYZE statement works similar to EXPLAIN, with the major difference being that it will actually execute the statement. This allows you to compare the estimates used as part of query planning to actual values encountered during execution. If the estimates differ significantly from the actual values, you should consider running ANALYZE TABLE on the affected tables.

Note:

When you use EXPLAIN ANALYZE to execute DML statements, modification to data is normally executed. Currently, the execution plan for DML statements **cannot** be shown yet.

11.5.2.49.1 Synopsis

```

ExplainSym ::=

  'EXPLAIN'
| 'DESCRIBE'
| 'DESC'

ExplainStmt ::=

  ExplainSym ( TableName ColumnName? | 'ANALYZE'? ExplainableStmt | 'FOR'
    ↪ 'CONNECTION' NUM | 'FORMAT' '=' ( stringLit | ExplainFormatType )
    ↪ ( 'FOR' 'CONNECTION' NUM | ExplainableStmt ) )

ExplainableStmt ::=

  SelectStmt
| DeleteFromStmt
| UpdateStmt
| InsertIntoStmt
| ReplaceIntoStmt
| UnionStmt

```

11.5.2.49.2 EXPLAIN ANALYZE output format

Different from EXPLAIN, EXPLAIN ANALYZE executes the corresponding SQL statement, records its runtime information, and returns the information together with the execution plan. Therefore, you can regard EXPLAIN ANALYZE as an extension of the EXPLAIN statement. Compared to EXPLAIN (for debugging query execution), the return results of EXPLAIN → ANALYZE also include columns of information such as `actRows`, `execution info`, `memory`, and `disk`. The details of these columns are shown as follows:

attribute name	description
<code>actRows</code>	Number of rows output by the operator.
<code>execution info</code>	Execution information of the operator. <code>time</code> represents the total <code>wall time</code> from entering the operator to leaving the operator, including the total execution time of all sub-operators. If the operator is called many times by the parent operator (in loops), then the time refers to the accumulated time. <code>loops</code> is the number of times the current operator is called by the parent operator.
<code>memory</code>	Memory space occupied by the operator.
<code>disk</code>	Disk space occupied by the operator.

11.5.2.49.3 Examples

```
CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT NOT
→ NULL);
```

```
Query OK, 0 rows affected (0.12 sec)
```

```
INSERT INTO t1 (c1) VALUES (1), (2), (3);
```

```
Query OK, 3 rows affected (0.02 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

```
EXPLAIN ANALYZE SELECT * FROM t1 WHERE id = 1;
```

```
+--
```

```
→ -----+-----+-----+-----+-----+
```

```

+----+-----+-----+-----+-----+
| id      | estRows | actRows | task | access object | execution info
+----+-----+-----+-----+-----+
| Point_Get_1 | 1.00   | 1       | root | table:t1    | time:757.205µs, loops:2,
|             |          |         |       | Get:{num_rpc:1, total_time:697.051µs} | handle:1 | N/A | N/A |
+----+-----+-----+-----+-----+
1 row in set (0.01 sec)

```

```
EXPLAIN ANALYZE SELECT * FROM t1;
```

```

+----+-----+-----+-----+-----+
| id      | estRows | actRows | task | access object | execution
|       | info    |         |       |               | info
+----+-----+-----+-----+-----+
|       | operator info           | memory   | disk  |
+----+-----+-----+-----+-----+
| TableReader_5 | 13.00  | 13     | root   |               | time:923.459µs
|             |          |         |       | , loops:2, cop_task: {num: 4, max: 839.788µs, min: 779.374µs, avg:
|             |          |         |       | 810.926µs, p95: 839.788µs, max_proc_keys: 12, p95_proc_keys: 12,
|             |          |         |       | rpc_num: 4, rpc_time: 3.116964ms, copr_cache_hit_ratio: 0.00} | data:
|             |          |         |       | TableFullScan_4 | 632 Bytes | N/A |
| -TableFullScan_4 | 13.00  | 13     | cop[tikv] | table:t1    | proc max:0s,
|             |          |         |       | min:0s, p80:0s, p95:0s, iters:4, tasks:4
|             |         |         |       | keep order:false, stats:pseudo | N/A | N/A |
+----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

```

11.5.2.49.4 Execution information of operators

In addition to the basic `time` and `loop` execution information, `execution info` also contains operator-specific execution information, which mainly includes the time consumed for the operator to send RPC requests and the duration of other steps.

`Point_Get`

The execution information from a `Point_Get` operator will typically contain the following information:

- `Get:{num_rpc:1, total_time:697.051µs}`: The number of the Get RPC requests (`num_rpc`) sent to TiKV and the total duration (`total_time`) of all RPC requests.
- `ResolveLock:{num_rpc:1, total_time:12.117495ms}`: If TiDB encounters a lock when reading data, it has to resolve the lock first, which generally occurs in the scenario of read-write conflict. This information indicates the duration of resolving locks.
- `regionMiss_backoff:{num:11, total_time:2010 ms},tikvRPC_backoff:{num ↵ :11, total_time:10691 ms}`: When an RPC request fails, TiDB will wait the backoff time before retrying the request. Backoff statistics include the type of backoff (such as `regionMiss` and `tikvRPC`), the total waiting time (`total_time`), and the total number of backoffs (`num`).

Batch_Point_Get

The execution information of the `Batch_Point_Get` operator is similar to that of the `Point_Get` operator, but `Batch_Point_Get` generally sends `BatchGet` RPC requests to TiKV to read data.

`BatchGet:{num_rpc:2, total_time:83.13µs}`: The number of RPC requests (`num_rpc`) of the `BatchGet` type sent to TiKV and the total time consumed (`total_time`) for all RPC requests.

TableReader

The execution information of a `TableReader` operator is typically as follows:

```
cop_task: {num: 6, max: 1.07587ms, min: 844.312µs, avg: 919.601µs, p95: ↵ 1.07587ms, max_proc_keys: 16, p95_proc_keys: 16, tot_proc: 1ms, ↵ tot_wait: 1ms, rpc_num: 6, rpc_time: 5.313996 ms, ↵ copr_cache_hit_ratio: 0.00}
```

- `cop_task`: Contains the execution information of `cop` tasks. For example:
 - `num`: The number of `cop` tasks.
 - `max`, `min`, `avg`, `p95`: The maximum, minimum, average, and P95 values of the execution time consumed for executing `cop` tasks.
 - `max_proc_keys` and `p95_proc_keys`: The maximum and P95 key-values scanned by TiKV in all `cop` tasks. If the difference between the maximum value and the P95 value is large, the data distribution might be imbalanced.
 - `rpc_num`, `rpc_time`: The total number and total time consumed for `Cop` RPC requests sent to TiKV.
 - `copr_cache_hit_ratio`: The hit rate of Coprocessor Cache for `cop` task requests. See [Coprocessor Cache Configuration](#) for details.
- `backoff`: Contains different types of backoff and the total waiting time of backoff.

Insert

The execution information of an `Insert` operator is typically as follows:

```
prepare:109.616µs, check_insert:{total_time:1.431678ms, mem_insert_time
    ↵ :667.878µs, prefetch:763.8µs, rpc:{BatchGet:{num_rpc:1, total_time
    ↵ :699.166µs},Get:{num_rpc:1, total_time:378.276µs }}}}
```

- `prepare`: The time consumed for preparing to write, including expression, default value and auto-increment value calculations.
- `check_insert`: This information generally appears in `insert ignore` and `insert ↵ on duplicate` statements, including conflict checking and the time consumed for writing data to TiDB transaction cache. Note that this time consumption does not include the time consumed for transaction commit. It contains the following information:
 - `total_time`: The total time spent on the `check_insert` step.
 - `mem_insert_time`: The time consumed for writing data to the TiDB transaction cache.
 - `prefetch`: The duration of retrieving the data that needs to be checked for conflicts from TiKV. This step sends a `Batch_Get` RPC request to TiKV to retrieve data.
 - `rpc`: The total time consumed for sending RPC requests to TiKV, which generally includes two types of RPC time, `BatchGet` and `Get`, among which:
 - * `BatchGet` RPC request is sent in the `prefetch` step.
 - * `Get` RPC request is sent when the `insert on duplicate` statement executes `duplicate update`.
- `backoff`: Contains different types of backoff and the total waiting time of backoff.

IndexJoin

The `IndexJoin` operator has 1 outer worker and N inner workers for concurrent execution. The join result preserves the order of the outer table. The detailed execution process is as follows:

1. The outer worker reads N outer rows, then wraps it into a task, and sends it to the result channel and the inner worker channel.
2. The inner worker receives the task, build key ranges from the task, and fetches inner rows according to the key ranges. It then builds the inner row hash table.
3. The main `IndexJoin` thread receives the task from the result channel and waits for the inner worker to finish handling the task.
4. The main `IndexJoin` thread joins each outer row by looking up to the inner rows' hash table.

The `IndexJoin` operator contains the following execution information:

```
inner:{total:4.297515932s, concurrency:5, task:17, construct:97.96291ms,
↪ fetch:4.164310088s, build:35.219574ms}, probe:53.574945ms
```

- **Inner:** The execution information of inner worker:
 - **total:** The total time consumed by the inner worker.
 - **concurrency:** The number of concurrent inner workers.
 - **task:** The total number of tasks processed by the inner worker.
 - **construct:** The preparation time before the inner worker reads the inner table rows corresponding to the task.
 - **fetch:** The total time consumed for it takes for the inner worker to read inner table rows.
 - **Build:** The total time consumed for it takes for the inner worker to construct the hash table of the corresponding inner table rows.
- **probe:** The total time consumed by the main `IndexJoin` thread to perform join operations with the hash table of the outer table rows and the inner table rows.

IndexHashJoin

The execution process of the `IndexHashJoin` operator is similar to that of the `IndexJoin` operator. `IndexHashJoin` operator also has 1 outer worker and N inner workers to execute in parallel, but the output order is not guaranteed to be consistent with that of the outer table. The detailed execution process is as follows:

1. The outer worker reads N outer rows, builds a task, and sends it to the inner worker channel.
2. The inner worker receives the tasks from the inner worker channel and performs the following three operations in order for every task:
 - a. Build a hash table from the outer rows
 - b. Build key ranges from outer rows and fetches inner rows
 - c. Probe the hash table and sends the join result to the result channel. Note: step a and step b are running concurrently.
3. The main thread of `IndexHashJoin` receives the join results from the result channel.

The `IndexHashJoin` operator contains the following execution information:

```
inner:{total:4.42922003s, concurrency:5, task:17, construct:96.207725ms,
↪ fetch:4.239324006s, build:24.567801ms, join:93.607362ms}
```

- **Inner:** the execution information of inner worker:

- **total**: the total time consumed by the inner worker.
- **concurrency**: the number of inner workers.
- **task**: The total number of tasks processed by the inner worker.
- **construct**: The preparation time before the inner worker reads the inner table rows.
- **fetch**: The total time consumed for inner worker to read inner table rows.
- **Build**: The total time consumed for inner worker to construct the hash table of the outer table rows.
- **join**: The total time consumed for inner worker to do join with the inner table rows and the hash table of outer table rows.

HashJoin

The `HashJoin` operator has an inner worker, an outer worker, and N join workers. The detailed execution process is as follows:

1. The inner worker reads inner table rows and constructs a hash table.
2. The outer worker reads the outer table rows, then wraps it into a task and sends it to the join worker.
3. The join worker waits for the hash table construction in step 1 to finish.
4. The join worker uses the outer table rows and hash table in the task to perform join operations, and then sends the join result to the result channel.
5. The main thread of `HashJoin` receives the join result from the result channel.

The `HashJoin` operator contains the following execution information:

```
build_hash_table:{total:146.071334ms, fetch:110.338509ms, build:35.732825ms
→ }, probe:{concurrency:5, total:857.162518ms, max:171.48271ms, probe
→ :125.341665ms, fetch:731.820853ms}
```

- **build_hash_table**: Reads the data of the inner table and constructs the execution information of the hash table:
 - **total**: The total time consumption.
 - **fetch**: The total time spent reading inner table data.
 - **build**: The total time spent constructing a hash table.
- **probe**: The execution information of join workers:
 - **concurrency**: The number of join workers.
 - **total**: The total time consumed by all join workers.
 - **max**: The longest time for a single join worker to execute.
 - **probe**: The total time consumed for joining with outer table rows and the hash table.
 - **fetch**: The total time that the join worker waits to read the outer table rows data.

lock_keys execution information

When a DML statement is executed in a pessimistic transaction, the execution information of the operator might also include the execution information of `lock_keys`. For example:

```
lock_keys: {time:94.096168ms, region:6, keys:8, lock_rpc:274.503214ms,
↪ rpc_count:6}
```

- `time`: The total duration of executing the `lock_keys` operation.
- `region`: The number of Regions involved in executing the `lock_keys` operation.
- `keys`: The number of Keys that need Lock.
- `lock_rpc`: The total time spent sending an RPC request of the `Lock` type to TiKV. Because multiple RPC requests can be sent in parallel, the total RPC time consumption might be greater than the total time consumption of the `lock_keys` operation.
- `rpc_count`: The total number of RPC requests of the `Lock` type sent to TiKV.

commit_txn execution information

When a write-type DML statement is executed in a transaction with `autocommit=1`, the execution information of the write operator will also include the duration information of the transaction commit. For example:

```
commit_txn: {prewrite:48.564544ms, wait_prewrite_binlog:47.821579,
↪ get_commit_ts:4.277455ms, commit:50.431774ms, region_num:7,
↪ write_keys:16, write_byte:536}
```

- `prewrite`: The time consumed for the `prewrite` phase of the 2PC commit of the transaction.
- `wait_prewrite_binlog`: The time consumed for waiting to write the prewrite Binlog.
- `get_commit_ts`: The time consumed for getting the transaction commit timestamp.
- `commit`: The time consumed for the `commit` phase during the 2PC commit of the transaction.
- `write_keys`: The total `keys` written in the transaction.
- `write_byte`: The total bytes of `key-value` written in the transaction, and the unit is byte.

11.5.2.49.5 MySQL compatibility

`EXPLAIN ANALYZE` is a feature of MySQL 8.0, but both the output format and the potential execution plans in TiDB differ substantially from MySQL.

11.5.2.49.6 See also

- [Understanding the Query Execution Plan](#)
- [EXPLAIN](#)
- [ANALYZE TABLE](#)
- [TRACE](#)

11.5.2.50 EXPLAIN

The `EXPLAIN` statement shows the execution plan for a query without executing it. It is complimented by `EXPLAIN ANALYZE` which will execute the query. If the output of `EXPLAIN` does not match the expected result, consider executing `ANALYZE TABLE` on each table in the query.

The statements `DESC` and `DESCRIBE` are aliases of this statement. The alternative usage of `EXPLAIN <tableName>` is documented under `SHOW [FULL] COLUMNS FROM`.

TiDB supports the `EXPLAIN [options] FOR CONNECTION connection_id` statement. However, this statement is different from the `EXPLAIN FOR` statement in MySQL. For more details, see [EXPLAIN FOR CONNECTION](#).

11.5.2.50.1 Synopsis

```

ExplainSym ::=

  'EXPLAIN'
| 'DESCRIBE'
| 'DESC'

ExplainStmt ::=

  ExplainSym ( TableName ColumnName? | 'ANALYZE'? ExplainableStmt | 'FOR'
    ↪ 'CONNECTION' NUM | 'FORMAT' '=' ( stringLit | ExplainFormatType )
    ↪ ( 'FOR' 'CONNECTION' NUM | ExplainableStmt ) )

ExplainableStmt ::=

  SelectStmt
| DeleteFromStmt
| UpdateStmt
| InsertIntoStmt
| ReplaceIntoStmt
| UnionStmt

```

11.5.2.50.2 EXPLAIN output format

Note:

When you use the MySQL client to connect to TiDB, to read the output result in a clearer way without line wrapping, you can use the pager `less ↪ -S` command. Then, after the `EXPLAIN` result is output, you can press the right arrow → button on your keyboard to horizontally scroll through the output.

Currently, EXPLAIN in TiDB outputs 5 columns: `id`, `estRows`, `task`, `access object`, `operator info`. Each operator in the execution plan is described by these attributes, with each row in the EXPLAIN output describing an operator. The description of each attribute is as follows:

Attribute	Description
name	
<code>id</code>	The operator ID is the unique identifier of the operator in the entire execution plan. In TiDB 2.1, the ID is formatted to display the tree structure of the operator. Data flows from the child node to the parent node. One and only one parent node for each operator.
<code>estRows</code>	The number of rows that the operator is expected to output. This number is estimated according to the statistics and the operator's logic. <code>estRows</code> is called <code>count</code> in the earlier versions of TiDB 4.0.
<code>task</code>	The type of task the operator belongs to. Currently, the execution plans are divided into two tasks: <code>root</code> task, which is executed on tidb-server, and <code>cop</code> task, which is performed in parallel on TiKV or TiFlash. The topology of the execution plan at the task level is that a root task followed by many cop tasks. The root task uses the output of cop tasks as input. The cop tasks refer to tasks that TiDB pushes down to TiKV or TiFlash. Each cop task is distributed in the TiKV cluster or the TiFlash cluster, and is executed by multiple processes.
<code>access object</code>	Data item information accessed by the operator. The information includes <code>table</code> , <code>partition</code> , and <code>index</code> (if any). Only operators that directly access the data have such information.
<code>operator info</code>	Other information about the operator. <code>operator info</code> of each operator is different. You can refer to the following examples.

11.5.2.50.3 Examples

```
EXPLAIN SELECT 1;
```

```
+-----+-----+-----+-----+
| id      | estRows | task   | access object | operator info |
+-----+-----+-----+-----+
| Projection_3 | 1.00   | root   |              | 1->Column#1  |
| -TableDual_4 | 1.00   | root   |              | rows:1        |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

```
CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT NOT
→ NULL);
```

```
Query OK, 0 rows affected (0.10 sec)
```

```
INSERT INTO t1 (c1) VALUES (1), (2), (3);
```

```
Query OK, 3 rows affected (0.02 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

```
EXPLAIN SELECT * FROM t1 WHERE id = 1;
```

id	estRows	task	access object	operator info
Point_Get_1	1.00	root	table:t1	handle:1

1 row in set (0.00 sec)

```
DESC SELECT * FROM t1 WHERE id = 1;
```

id	estRows	task	access object	operator info
Point_Get_1	1.00	root	table:t1	handle:1

1 row in set (0.00 sec)

```
DESCRIBE SELECT * FROM t1 WHERE id = 1;
```

id	estRows	task	access object	operator info
Point_Get_1	1.00	root	table:t1	handle:1

1 row in set (0.00 sec)

```
EXPLAIN INSERT INTO t1 (c1) VALUES (4);
```

id	estRows	task	access object	operator info
Insert_1	N/A	root		N/A

1 row in set (0.00 sec)

```
EXPLAIN UPDATE t1 SET c1=5 WHERE c1=3;
```

```
+--+
| id          | estRows | task      | access object | operator info
+--+
| Update_4    | N/A     | root      |               | N/A
| -TableReader_8 | 0.00   | root      |               | data:
|   -Selection_7 | 0.00   | cop[tikv] | eq(test.t1.c1,
|     3)         |         |           | keep order:
|       -TableFullScan_6 | 3.00   | cop[tikv] | table:t1 | keep order:
|         | false, stats:pseudo |
+--+
4 rows in set (0.00 sec)
```

EXPLAIN DELETE FROM t1 WHERE c1=3;

```
+--+
| id          | estRows | task      | access object | operator info
+--+
| Delete_4    | N/A     | root      |               | N/A
| -TableReader_8 | 0.00   | root      |               | data:
|   -Selection_7 | 0.00   | cop[tikv] | eq(test.t1.c1,
|     3)         |         |           | keep order:
|       -TableFullScan_6 | 3.00   | cop[tikv] | table:t1 | keep order:
|         | false, stats:pseudo |
+--+
4 rows in set (0.01 sec)
```

If you do not specify the `FORMAT`, or specify `FORMAT = "row"`, `EXPLAIN` statement will output the results in a tabular format. See [Understand the Query Execution Plan](#) for more information.

In addition to the MySQL standard result format, TiDB also supports DotGraph and you need to specify `FORMAT = "dot"` as in the following example:

```
create table t(a bigint, b bigint);
desc format = "dot" select A.a, B.b from t A join t B on A.a > B.b where A.
    ↪ a < 10;
```

```
+--+
    ↪ -----
    ↪
    ↪ |
| dot contents
    ↪
    ↪ |
+--+
    ↪ -----
    ↪
    ↪ |
|
digraph Projection_8 {
subgraph cluster8{
node [style=filled, color=lightgrey]
color=black
label = "root"
"Projection_8" -> "HashJoin_9"
"HashJoin_9" -> "TableReader_13"
"HashJoin_9" -> "Selection_14"
"Selection_14" -> "TableReader_17"
}
subgraph cluster12{
node [style=filled, color=lightgrey]
color=black
label = "cop"
"Selection_12" -> "TableFullScan_11"
}
subgraph cluster16{
node [style=filled, color=lightgrey]
color=black
label = "cop"
"Selection_16" -> "TableFullScan_15"
}
"TableReader_13" -> "Selection_12"
"TableReader_17" -> "Selection_16"
}
```

```
|  
+--  
  ↵  
  ↵  
1 row in set (0.00 sec)
```

If the `dot` program (in the `graphviz` package) is installed on your computer, you can generate a PNG file using the following method:

```
dot xx.dot -T png -o
```

The `xx.dot` is the result returned by the above statement.

If the `dot` program is not installed on your computer, copy the result to [this website](#) to get a tree diagram:

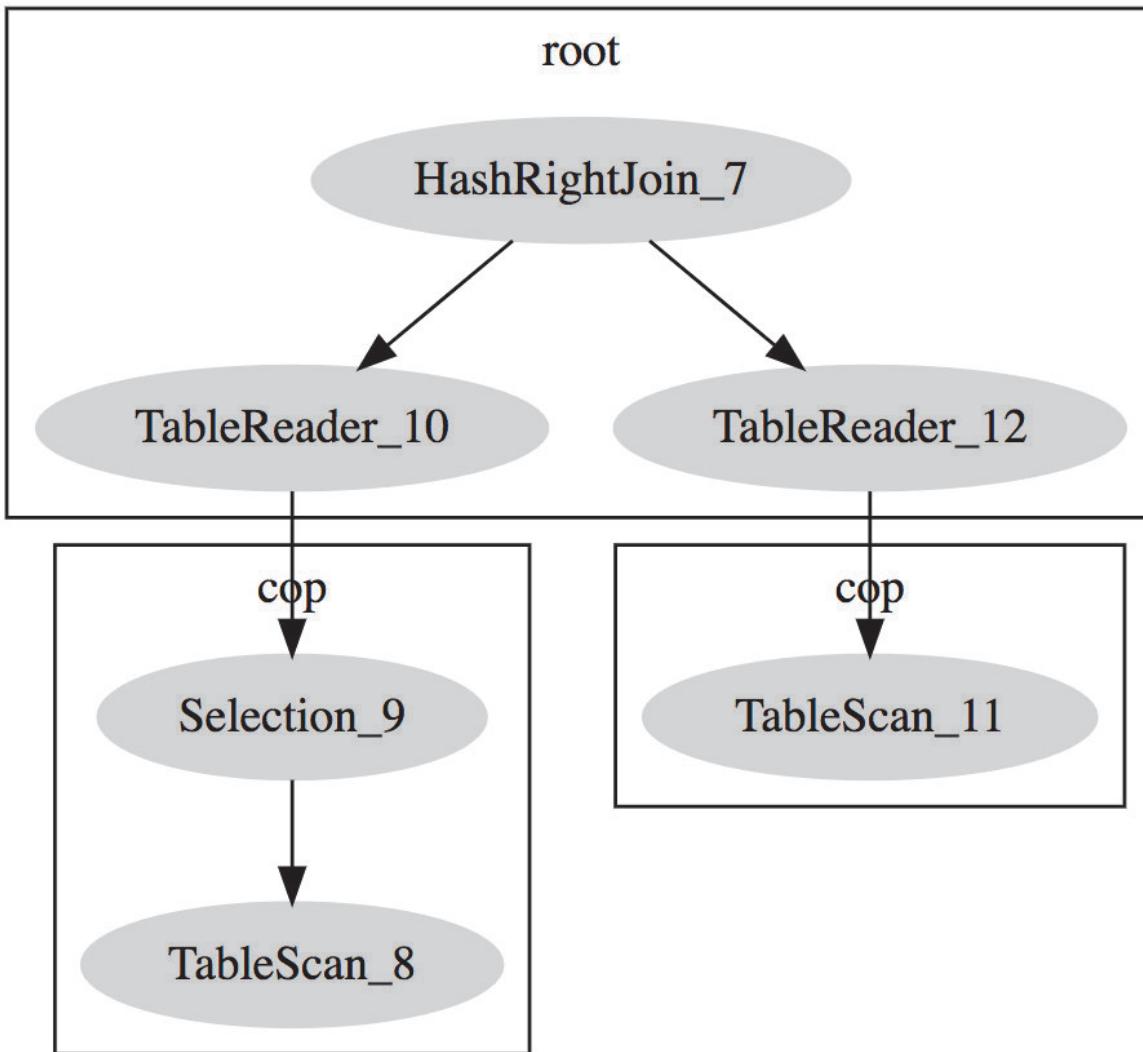


Figure 201: Explain Dot

11.5.2.50.4 MySQL compatibility

- Both the format of EXPLAIN and the potential execution plans in TiDB differ substantially from MySQL.
- TiDB does not support the FORMAT=JSON or FORMAT=TREE options.
- TiDB does not currently support EXPLAIN for insert statements.

EXPLAIN FOR CONNECTION

EXPLAIN FOR CONNECTION is used to get the execution plan of the currently executed SQL query or the last executed SQL query in a connection. The output format is the same

as that of EXPLAIN. However, the implementation of EXPLAIN FOR CONNECTION in TiDB is different from that in MySQL. Their differences (apart from the output format) are listed as follows:

- MySQL returns the query plan that is **being executing**, while TiDB returns the **last executed** query plan.
- MySQL requires the login user to be the same as the connection being queried, or the login user has the **PROCESS** privilege; while TiDB requires the login user to be the same as the connection being queried, or the login user has the **SUPER** privilege.

11.5.2.50.5 See also

- Understanding the Query Execution Plan
- EXPLAIN ANALYZE
- ANALYZE TABLE
- TRACE

11.5.2.51 FLASHBACK TABLE

The FLASHBACK TABLE syntax is introduced since TiDB 4.0. You can use the FLASHBACK → TABLE statement to restore the tables and data dropped by the DROP or TRUNCATE operation within the Garbage Collection (GC) lifetime.

The system variable `tidb_gc_life_time` (default: 10m0s) defines the retention time of earlier versions of rows. The current `safePoint` of where garbage collection has been performed up to can be obtained with the following query:

```
SELECT * FROM mysql.tidb WHERE variable_name = 'tikv_gc_safe_point';
```

As long as the table is dropped by DROP or TRUNCATE statements after the `tikv_gc_safe_point` time, you can restore the table using the FLASHBACK TABLE statement.

11.5.2.51.1 Syntax

```
FLASHBACK TABLE table_name [TO other_table_name]
```

11.5.2.51.2 Synopsis

```
FlashbackTableStmt ::=  
    'FLASHBACK' 'TABLE' TableName FlashbackToNewName  
  
TableName ::=  
    Identifier ('.' Identifier)?  
  
FlashbackToNewName ::=  
    ('TO' Identifier)?
```

11.5.2.51.3 Notes

If a table is dropped and the GC lifetime has passed, you can no longer use the `FLASHBACK TABLE` statement to recover the dropped data. Otherwise, an error like `Can't find dropped / truncated table 't' in GC safe point 2020-03-16 16:34:52 +0800 CST` will be returned.

Pay attention to the following conditions and requirements when you enable TiDB Binlog and use the `FLASHBACK TABLE` statement:

- The downstream secondary cluster must also support `FLASHBACK TABLE`.
- The GC lifetime of the secondary cluster must be longer than that of the primary cluster.
- The delay of replication between the upstream and downstream might also cause the failure to recover data to the downstream.
- If an error occurs when TiDB Binlog is replicating a table, you need to filter that table in TiDB Binlog and manually import all data of that table.

11.5.2.51.4 Example

- Recover the table data dropped by the `DROP` operation:

```
DROP TABLE t;
```

```
FLASHBACK TABLE t;
```

- Recover the table data dropped by the `TRUNCATE` operation. Because the truncated table `t` still exists, you need to rename the table `t` to be recovered. Otherwise, an error will be returned because the table `t` already exists.

```
TRUNCATE TABLE t;
```

```
FLASHBACK TABLE t TO t1;
```

11.5.2.51.5 Implementation principle

When deleting a table, TiDB only deletes the table metadata, and writes the table data (row data and index data) to be deleted to the `mysql.gc_delete_range` table. The GC Worker in the TiDB background periodically removes from the `mysql.gc_delete_range` table the keys that exceed the GC lifetime.

Therefore, to recover a table, you only need to recover the table metadata and delete the corresponding row record in the `mysql.gc_delete_range` table before the GC Worker deletes the table data. You can use a snapshot read of TiDB to recover the table metadata. For details of snapshot read, refer to [Read Historical Data](#).

The following is the working process of `FLASHBACK TABLE t TO t1`:

1. TiDB searches the recent DDL history jobs and locates the first DDL operation of the `DROP TABLE` or the `truncate table` type on table `t`. If TiDB fails to locate one, an error is returned.
2. TiDB checks whether the starting time of the DDL job is before `tikv_gc_safe_point`. If it is before `tikv_gc_safe_point`, it means that the table dropped by the `DROP` or `TRUNCATE` operation has been cleaned up by the GC and an error is returned.
3. TiDB uses the starting time of the DDL job as the snapshot to read historical data and read table metadata.
4. TiDB deletes GC tasks related to table `t` in `mysql.gc_delete_range`.
5. TiDB changes `name` in the table's metadata to `t1`, and uses this metadata to create a new table. Note that only the table name is changed but not the table ID. The table ID is the same as that of the previously dropped table `t`.

From the above process, you can see that TiDB always operates on the metadata of the table, and the user data of the table has never been modified. The restored table `t1` has the same ID as the previously dropped table `t`, so `t1` can read the user data of `t`.

Note:

You cannot use `FLASHBACK` statements to restore the same deleted table multiple times, because the ID of the restored table is the same ID of the dropped table, and TiDB requires that all existing tables must have a globally unique table ID.

The `FLASHBACK TABLE` operation is done by TiDB obtaining the table metadata through snapshot read, and then going through the process of table creation similar to `CREATE TABLE`. Therefore, `FLASHBACK TABLE` is, in essence, a kind of DDL operation.

11.5.2.51.6 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.52 FLUSH PRIVILEGES

This statement triggers TiDB to reload the in-memory copy of privileges from the privilege tables. You should execute `FLUSH PRIVILEGES` after making manual edits to tables such as `mysql.user`. Executing this statement is not required after using privilege statements such as `GRANT` or `REVOKE`. Executing this statement requires the `RELOAD` privilege.

11.5.2.52.1 Synopsis

```

FlushStmt ::=

  'FLUSH' NoWriteToBinLogAliasOpt FlushOption

NoWriteToBinLogAliasOpt ::=

  ( 'NO_WRITE_TO_BINLOG' | 'LOCAL' )?

FlushOption ::=

  'PRIVILEGES'
| 'STATUS'
| 'TIDB' 'PLUGINS' PluginNameList
| 'HOSTS'
| LogTypeOpt 'LOGS'
| TableOrTables TableNameListOpt WithReadLockOpt

```

11.5.2.52.2 Examples

```

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)

```

11.5.2.52.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be [reported via an issue](#) on GitHub.

11.5.2.52.4 See also

- [Privilege Management](#)

11.5.2.53 FLUSH STATUS

This statement is included for compatibility with MySQL. It has no effect on TiDB, which uses Prometheus and Grafana for centralized metrics collection instead of `SHOW STATUS`.

11.5.2.53.1 Synopsis

```

FlushStmt ::=

  'FLUSH' NoWriteToBinLogAliasOpt FlushOption

NoWriteToBinLogAliasOpt ::=

  ( 'NO_WRITE_TO_BINLOG' | 'LOCAL' )?

FlushOption ::=

  'PRIVILEGES'

```

```

| 'STATUS'
| 'TIDB' 'PLUGINS' PluginNameList
| 'HOSTS'
| LogTypeOpt 'LOGS'
| TableOrTables TableNameListOpt WithReadLockOpt

```

11.5.2.53.2 Examples

```

mysql> show status;
+-----+-----+
| Variable_name | Value |
+-----+-----+
| Ssl_cipher_list |      |
| server_id | 93e2e07d-6bb4-4a1b-90b7-e035fae154fe |
| ddl_schema_version | 141 |
| Ssl_verify_mode | 0 |
| Ssl_version |      |
| Ssl_cipher |      |
+-----+
6 rows in set (0.01 sec)

mysql> show global status;
+-----+-----+
| Variable_name | Value |
+-----+-----+
| Ssl_cipher |      |
| Ssl_cipher_list |      |
| Ssl_verify_mode | 0 |
| Ssl_version |      |
| server_id | 93e2e07d-6bb4-4a1b-90b7-e035fae154fe |
| ddl_schema_version | 141 |
+-----+
6 rows in set (0.00 sec)

mysql> flush status;
Query OK, 0 rows affected (0.00 sec)

mysql> show status;
+-----+-----+
| Variable_name | Value |
+-----+-----+
| Ssl_cipher |      |
| Ssl_cipher_list |      |
| Ssl_verify_mode | 0 |

```

```

| Ssl_version      |
| ddl_schema_version | 141
| server_id        | 93e2e07d-6bb4-4a1b-90b7-e035fae154fe |
+-----+
6 rows in set (0.00 sec)

```

11.5.2.53.3 MySQL compatibility

- This statement is included only for compatibility with MySQL.

11.5.2.53.4 See also

- [SHOW \[GLOBAL|SESSION\] STATUS](#)

11.5.2.54 FLUSH TABLES

This statement is included for compatibility with MySQL. It has no effective usage in TiDB.

11.5.2.54.1 Synopsis

```

FlushStmt ::=

  'FLUSH' NoWriteToBinLogAliasOpt FlushOption

NoWriteToBinLogAliasOpt ::=
  ( 'NO_WRITE_TO_BINLOG' | 'LOCAL' )?

FlushOption ::=
  'PRIVILEGES'
| 'STATUS'
| 'TIDB' 'PLUGINS' PluginNameList
| 'HOSTS'
| LogTypeOpt 'LOGS'
| TableOrTables TableNameListOpt WithReadLockOpt

LogTypeOpt ::=
  ( 'BINARY' | 'ENGINE' | 'ERROR' | 'GENERAL' | 'SLOW' )?

TableOrTables ::=
  'TABLE'
| 'TABLES'

TableNameListOpt ::=

```

```

TableNameList?

WithReadLockOpt ::=

( 'WITH' 'READ' 'LOCK' )?

```

11.5.2.54.2 Examples

```

mysql> FLUSH TABLES;
Query OK, 0 rows affected (0.00 sec)

mysql> FLUSH TABLES WITH READ LOCK;
ERROR 1105 (HY000): FLUSH TABLES WITH READ LOCK is not supported. Please
→ use @@tidb_snapshot

```

11.5.2.54.3 MySQL compatibility

- TiDB does not have a concept of table cache as in MySQL. Thus, `FLUSH TABLES` is parsed but ignored in TiDB for compatibility.
- The statement `FLUSH TABLES WITH READ LOCK` produces an error, as TiDB does not currently support locking tables. It is recommended to use [Historical reads](#) for this purpose instead.

11.5.2.54.4 See also

- [Read historical data](#)

11.5.2.55 GRANT <privileges>

This statement allocates privileges to a pre-existing user in TiDB. The privilege system in TiDB follows MySQL, where credentials are assigned based on a database/table pattern. Executing this statement requires the `GRANT OPTION` privilege and all privileges you allocate.

11.5.2.55.1 Synopsis

```

GrantStmt ::=

'GRANT' PrivElemList 'ON' ObjectType PrivLevel 'TO' UserSpecList
    → RequireClauseOpt WithGrantOptionOpt

PrivElemList ::=

PrivElem ( ',' PrivElem )*

PrivElem ::=

PrivType ( '(' ColumnNameList ')' )?

```

```

PrivType ::=

  'ALL' 'PRIVILEGES'?
|  'ALTER' 'ROUTINE'?
|  'CREATE' ( 'USER' | 'TEMPORARY' 'TABLES' | 'VIEW' | 'ROLE' | 'ROUTINE' )
  ↵ ?
|  'TRIGGER'
|  'DELETE'
|  'DROP' 'ROLE'?
|  'PROCESS'
|  'EXECUTE'
|  'INDEX'
|  'INSERT'
|  'SELECT'
|  'SUPER'
|  'SHOW' ( 'DATABASES' | 'VIEW' )
|  'UPDATE'
|  'GRANT' 'OPTION'
|  'REFERENCES'
|  'REPLICATION' ( 'SLAVE' | 'CLIENT' )
|  'USAGE'
|  'RELOAD'
|  'FILE'
|  'CONFIG'
|  'LOCK' 'TABLES'
|  'EVENT'
|  'SHUTDOWN'

ObjectType ::=
  'TABLE'?

PrivLevel ::=
  '*' ( '.' '*' )?
|  Identifier ( '.' ( '*' | Identifier ) )?

UserSpecList ::=
  UserSpec ( ',' UserSpec )*

```

11.5.2.55.2 Examples

```
mysql> CREATE USER 'newuser' IDENTIFIED BY 'mypassword';
Query OK, 1 row affected (0.02 sec)
```

```
mysql> GRANT ALL ON test.* TO 'newuser';
```

```
Query OK, 0 rows affected (0.03 sec)

mysql> SHOW GRANTS FOR 'newuser';
+-----+
| Grants for newuser@% |
+-----+
| GRANT USAGE ON *.* TO 'newuser'@'%' |
| GRANT ALL PRIVILEGES ON test.* TO 'newuser'@'%' |
+-----+
2 rows in set (0.00 sec)
```

11.5.2.55.3 MySQL compatibility

- Similar to MySQL, the USAGE privilege denotes the ability to log into a TiDB server.
- Column level privileges are not currently supported.
- Similar to MySQL, when the NO_AUTO_CREATE_USER sql mode is not present, the GRANT statement will automatically create a new user with an empty password when a user does not exist. Removing this sql-mode (it is enabled by default) presents a security risk.

11.5.2.55.4 See also

- [GRANT <role>](#)
- [REVOKE <privileges>](#)
- [SHOW GRANTS](#)
- [Privilege Management](#)

11.5.2.56 GRANT <role>

Assigns a previously created role to an existing user. The user can use then use the statement SET ROLE <rolename> to assume the privileges of the role, or SET ROLE ALL to assume all roles that have been assigned.

11.5.2.56.1 Synopsis

```
GrantRoleStmt ::=  
    'GRANT' RolenameList 'TO' UsernameList  
  
RolennameList ::=  
    Rolename ( ',' Rolename )*  
  
UsernameList ::=  
    Username ( ',' Username )*
```

11.5.2.56.2 Examples

Create a new role for the analytics team, and a new user called jennifer:

```
$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 37
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
  ↪ License 2.0) Community Edition, MySQL 5.7 compatible

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> CREATE ROLE analyticsteam;
Query OK, 0 rows affected (0.02 sec)

mysql> GRANT SELECT ON test.* TO analyticsteam;
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE USER jennifer;
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT analyticsteam TO jennifer;
Query OK, 0 rows affected (0.01 sec)
```

Note that by default jennifer needs to SET ROLE analyticsteam in order to be able to use the privileges associated with the role:

```
$ mysql -ujennifer
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 32
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
  ↪ License 2.0) Community Edition, MySQL 5.7 compatible

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> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
2 rows in set (0.00 sec)

mysql> SHOW TABLES in test;
ERROR 1044 (42000): Access denied for user 'jennifer'@'%' to database 'test'
mysql> SET ROLE analyticsteam;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT Select ON test.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
3 rows in set (0.00 sec)

mysql> SHOW TABLES IN test;
+-----+
| Tables_in_test |
+-----+
| t1           |
+-----+
1 row in set (0.00 sec)
```

The statement `SET DEFAULT ROLE` can be used to associate a role to `jennifer` so that she will not have to execute the statement `SET ROLE` in order to assume the privileges associated with the role:

```
$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 34
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible
```

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> SET DEFAULT ROLE analyticsteam TO jennifer;
Query OK, 0 rows affected (0.02 sec)
```

```
$ mysql -ujennifer
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 35
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
    ↵ License 2.0) Community Edition, MySQL 5.7 compatible
```

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> SHOW GRANTS;
+-----+
| Grants for User          |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT Select ON test.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
3 rows in set (0.00 sec)
```

```
mysql> SHOW TABLES IN test;
+-----+
| Tables_in_test |
+-----+
| t1           |
```

```
+-----+
1 row in set (0.00 sec)
```

11.5.2.56.3 MySQL compatibility

This statement is understood to be fully compatible with roles, which are a feature of MySQL 8.0. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.56.4 See also

- [GRANT <privileges>](#)
- [CREATE ROLE](#)
- [DROP ROLE](#)
- [REVOKE <role>](#)
- [SET ROLE](#)
- [SET DEFAULT ROLE](#)
- [Role-Based Access Control](#)

11.5.2.57 INSERT

This statement inserts new rows into a table.

11.5.2.57.1 Synopsis

```
InsertIntoStmt ::=

  'INSERT' TableOptimizerHints PriorityOpt IgnoreOptional IntoOpt
    ↪ TableName PartitionNameListOpt InsertValues OnDuplicateKeyUpdate

TableOptimizerHints ::=

  hintComment?

PriorityOpt ::=

  ('LOW_PRIORITY' | 'HIGH_PRIORITY' | 'DELAYED')?

IgnoreOptional ::=

  'IGNORE'?

IntoOpt ::= 'INTO'?

TableName ::=

  Identifier ('.' Identifier)?

PartitionNameListOpt ::=

  ('PARTITION' '(' Identifier (',' Identifier)* ')')?
```

```

InsertValues ::=

  '(' ( ColumnNameListOpt ')' ( ValueSym ValuesList | SelectStmt | '('
    ↪ SelectStmt ')' | UnionStmt ) | SelectStmt ')' )

| ValueSym ValuesList
| SelectStmt
| UnionStmt
| 'SET' ColumnSetValue? ( ',' ColumnSetValue )*

OnDuplicateKeyUpdate ::=

  ( 'ON' 'DUPLICATE' 'KEY' 'UPDATE' AssignmentList )?

```

11.5.2.57.2 Examples

```

mysql> CREATE TABLE t1 (a INT);
Query OK, 0 rows affected (0.11 sec)

mysql> CREATE TABLE t2 LIKE t1;
Query OK, 0 rows affected (0.11 sec)

mysql> INSERT INTO t1 VALUES (1);
Query OK, 1 row affected (0.02 sec)

mysql> INSERT INTO t1 (a) VALUES (1);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO t2 SELECT * FROM t1;
Query OK, 2 rows affected (0.01 sec)
Records: 2 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM t1;
+---+
| a |
+---+
| 1 |
| 1 |
+---+
2 rows in set (0.00 sec)

mysql> SELECT * FROM t2;
+---+
| a |
+---+
| 1 |

```

```

|   1 |
+-----+
2 rows in set (0.00 sec)

mysql> INSERT INTO t2 VALUES (2),(3),(4);
Query OK, 3 rows affected (0.02 sec)
Records: 3 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM t2;
+-----+
| a   |
+-----+
|   1 |
|   1 |
|   2 |
|   3 |
|   4 |
+-----+
5 rows in set (0.00 sec)

```

11.5.2.57.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.57.4 See also

- [DELETE](#)
- [SELECT](#)
- [UPDATE](#)
- [REPLACE](#)

11.5.2.58 KILL TiDB

The statement KILL TiDB is used to terminate connections in TiDB.

11.5.2.58.1 Synopsis

KillStmt ::= KillOrKillTiDB ('CONNECTION' 'QUERY')? NUM
KillOrKillTiDB ::= 'KILL' 'TiDB'?

11.5.2.58.2 Examples

```
mysql> SHOW PROCESSLIST;
+----+-----+-----+-----+-----+-----+-----+
| Id | User | Host      | db   | Command | Time | State    | Info          |
+----+-----+-----+-----+-----+-----+-----+
| 1  | root | 127.0.0.1 | test | Query   | 0    | 2        | SHOW PROCESSLIST |
| 2  | root | 127.0.0.1 |     | Sleep   | 4    | 2        |                 |
+----+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

KILL TIDB 2;
Query OK, 0 rows affected (0.00 sec)
```

11.5.2.58.3 MySQL compatibility

- By design, KILL is not compatible with MySQL by default. This helps prevent against a case of a connection being terminated on the wrong TiDB server, because it is common to place multiple TiDB servers behind a load balancer.
- DO NOT set `compatible-kill-query = true` in your configuration file UNLESS you are certain that clients will be always connected to the same TiDB node. This is because pressing `ctrl+c` in the default MySQL client opens a new connection in which KILL is executed. If there are proxies in between, the new connection might be routed to a different TiDB node, which possibly kills a different session.
- The `KILL TIDB` statement is a TiDB extension. The feature of this statement is similar to the MySQL `KILL [CONNECTION|QUERY]` command and the MySQL command-line `ctrl+c` feature. It is safe to use `KILL TIDB` on the same TiDB node.

11.5.2.58.4 See also

- [SHOW \[FULL\] PROCESSLIST](#)
- [CLUSTER_PROCESSLIST](#)

11.5.2.59 LOAD DATA

The `LOAD DATA` statement batch loads data into a TiDB table.

11.5.2.59.1 Synopsis

```
LoadDataStmt ::=  
  'LOAD' 'DATA' LocalOpt 'INFILE' stringLit DuplicateOpt 'INTO' 'TABLE'  
    ↪ TableName CharsetOpt Fields Lines IgnoreLines  
    ↪ ColumnNameOrUserVarListOptWithBrackets LoadDataSetSpecOpt
```

11.5.2.59.2 Parameters

LocalOpt

You can specify that the imported data file is located on the client or on the server by configuring the `LocalOpt` parameter. Currently, TiDB only supports data import from the client. Therefore, when importing data, set the value of `LocalOpt` to `Local`.

Fields and Lines

You can specify how to process the data format by configuring the `Fields` and `Lines` parameters.

- `FIELDS TERMINATED BY`: Specifies the separating character of each data.
- `FIELDS ENCLOSED BY`: Specifies the enclosing character of each data.
- `LINES TERMINATED BY`: Specifies the line terminator, if you want to end a line with a certain character.

Take the following data format as an example:

```
"bob","20","street 1"\r\n  
"alice","33","street 1"\r\n
```

If you want to extract `bob`, `20`, and `street 1`, specify the separating character as `' , '`, and the enclosing character as `'\"'`:

```
FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\r\n'
```

If you do not specify the parameters above, the imported data is processed in the following way by default:

```
FIELDS TERMINATED BY '\t' ENCLOSED BY ''  
LINES TERMINATED BY '\n'
```

IGNORE number LINES

You can ignore the first `number` lines of a file by configuring the `IGNORE number LINES` parameter. For example, if you configure `IGNORE 1 LINES`, the first line of a file is ignored.

In addition, TiDB currently only supports parsing the syntax of the `DuplicateOpt`, `CharsetOpt`, and `LoadDataSetSpecOpt` parameters.

11.5.2.59.3 Examples

```
CREATE TABLE trips (
    trip_id bigint NOT NULL PRIMARY KEY AUTO_INCREMENT,
    duration integer not null,
    start_date datetime,
    end_date datetime,
    start_station_number integer,
    start_station varchar(255),
    end_station_number integer,
    end_station varchar(255),
    bike_number varchar(255),
    member_type varchar(255)
);
```

```
Query OK, 0 rows affected (0.14 sec)
```

The following example imports data using LOAD DATA. Comma is specified as the separating character. The double quotation marks that enclose the data is ignored. The first line of the file is ignored.

If you see the error message `ERROR 1148 (42000): the used command is not allowed with this TiDB version`, refer to [ERROR 1148 \(42000\): the used command is not allowed with this TiDB version](#).

```
LOAD DATA LOCAL INFILE '/mnt/evo970/data-sets/bikeshare-data/2017Q4-
→ capitalbikeshare-tripdata.csv' INTO TABLE trips FIELDS TERMINATED BY
→ ',' ENCLOSED BY '\"' LINES TERMINATED BY '\r\n' IGNORE 1 LINES (
→ duration, start_date, end_date, start_station_number, start_station,
→ end_station_number, end_station, bike_number, member_type);
```

```
Query OK, 815264 rows affected (39.63 sec)
Records: 815264 Deleted: 0 Skipped: 0 Warnings: 0
```

LOAD DATA also supports using hexadecimal ASCII character expressions or binary ASCII character expressions as the parameters for FIELDS ENCLOSED BY and FIELDS TERMINATED BY. See the following example:

```
LOAD DATA LOCAL INFILE '/mnt/evo970/data-sets/bikeshare-data/2017Q4-
→ capitalbikeshare-tripdata.csv' INTO TABLE trips FIELDS TERMINATED BY
→ x'2c' ENCLOSED BY b'100010' LINES TERMINATED BY '\r\n' IGNORE 1 LINES
→ (duration, start_date, end_date, start_station_number, start_station
→ , end_station_number, end_station, bike_number, member_type);
```

In the above example, `x'2c'` is the hexadecimal representation of the `,` character and `b'100010'` is the binary representation of the `"` character.

11.5.2.59.4 MySQL compatibility

This statement is understood to be fully compatible with MySQL except for the `LOAD DATA...REPLACE INTO` syntax [#24515](#). Any other compatibility differences should be reported via an issue on GitHub.

Note:

In earlier releases of TiDB, `LOAD DATA` committed every 20000 rows. By default, TiDB now commits all rows in one transaction. This can result in the error `ERROR 8004 (HY000) at line 1: Transaction is too large`, `size: 100000058` after upgrading from TiDB 4.0 or earlier versions.

The recommended way to resolve this error is to increase the `txn-total-size-limit` value in your `tidb.toml` file. If you are unable to increase this limit, you can also restore the previous behavior by setting `tidb_dml_batch_size` to 20000.

11.5.2.59.5 See also

- [INSERT](#)
- [Import Example Database](#)
- [TiDB Lightning](#)

11.5.2.60 LOAD STATS

The `LOAD STATS` statement is used to load the statistics into TiDB.

11.5.2.60.1 Synopsis

```
LoadStatsStmt ::=  
  'LOAD' 'STATS' stringLit
```

11.5.2.60.2 Examples

You can access the address `http://${tidb-server-ip}:${tidb-server-status-port}/stats/dump/${db_name}/${table_name}` to download the TiDB instance's statistics.

You can also use `LOAD STATS ${stats_path}` to load the specific statistics file.

The `${stats_path}` can be an absolute path or a relative path. If you use a relative path, the corresponding file is found starting from the path where `tidb-server` is started. Here is an example:

```
LOAD STATS '/tmp/stats.json';
```

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

11.5.2.60.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.60.4 See also

- [Statistics](#)

11.5.2.61 MODIFY COLUMN

The `ALTER TABLE.. MODIFY COLUMN` statement modifies a column on an existing table. The modification can include changing the data type and attributes. To rename at the same time, use the `CHANGE COLUMN` statement instead.

Since v5.1.0, TiDB has supported changes of data types for Reorg data, including but not limited to:

- Changing `VARCHAR` to `BIGINT`
- Modifying the `DECIMAL` precision
- Compressing the length of `VARCHAR(10)` to `VARCHAR(5)`

11.5.2.61.1 Synopsis

```
AlterTableStmt ::=  
  'ALTER' IgnoreOptional 'TABLE' TableName ( AlterTableSpecListOpt  
    ↪ AlterTablePartitionOpt | 'ANALYZE' 'PARTITION' PartitionNameList (|  
    ↪ 'INDEX' IndexNameList )? AnalyzeOptionListOpt )  
  
AlterTableSpec ::=  
  TableOptionList  
| 'SET' 'TIFLASH' 'REPLICA' LengthNum LocationLabelList  
| 'CONVERT' 'TO' CharsetKw ( CharsetName | 'DEFAULT' ) OptCollate  
| 'ADD' ( ColumnKeywordOpt IfNotExists ( ColumnDef ColumnPosition | '()' |  
    ↪ TableElementList ')' ) | Constraint | 'PARTITION' IfNotExists  
    ↪ NoWriteToBinLogAliasOpt ( PartitionDefinitionListOpt | 'PARTITIONS'  
    ↪ NUM ) )  
| ( ( 'CHECK' | 'TRUNCATE' ) 'PARTITION' | ( 'OPTIMIZE' | 'REPAIR' | '|'  
    ↪ REBUILD' ) 'PARTITION' NoWriteToBinLogAliasOpt )  
    ↪ AllOrPartitionNameList  
| 'COALESCE' 'PARTITION' NoWriteToBinLogAliasOpt NUM
```

```

|   'DROP' ( ColumnKeywordOpt IfExists ColumnName RestrictOrCascadeOpt | '
|     ↳ PRIMARY' 'KEY' | 'PARTITION' IfExists PartitionNameList | (
|       ↳ KeyOrIndex IfExists | 'CHECK' ) Identifier | 'FOREIGN' 'KEY' IfExists
|       ↳ Symbol )
|   'EXCHANGE' 'PARTITION' Identifier 'WITH' 'TABLE' TableName
|     ↳ WithValidationOpt
|   ( 'IMPORT' | 'DISCARD' ) ( 'PARTITION' AllOrPartitionNameList )? '
|     ↳ TABLESPACE'
|   'REORGANIZE' 'PARTITION' NoWriteToBinLogAliasOpt
|     ↳ ReorganizePartitionRuleOpt
|   'ORDER' 'BY' AlterOrderItem ( ',' AlterOrderItem )*
|   ( 'DISABLE' | 'ENABLE' ) 'KEYS'
|   ( 'MODIFY' ColumnKeywordOpt IfExists | 'CHANGE' ColumnKeywordOpt
|     ↳ IfExists ColumnName ) ColumnDef ColumnPosition
|   'ALTER' ( ColumnKeywordOpt ColumnName ( 'SET' 'DEFAULT' ( SignedLiteral
|     ↳ | '(' Expression ')' ) | 'DROP' 'DEFAULT' ) | 'CHECK' Identifier
|     ↳ EnforcedOrNot | 'INDEX' Identifier IndexInvisible )
|   'RENAME' ( ( 'COLUMN' | KeyOrIndex ) Identifier 'TO' Identifier | ( 'TO'
|     ↳ | '='? | 'AS' ) TableName )
|   LockClause
|   AlgorithmClause
|   'FORCE'
|   ( 'WITH' | 'WITHOUT' ) 'VALIDATION'
|   'SECONDARY_LOAD'
|   'SECONDARY_UNLOAD'

```

ColumnKeywordOpt ::= 'COLUMN'?

ColumnDef ::=
 ColumnName (Type | 'SERIAL') ColumnOptionListOpt

ColumnPosition ::=
 ('FIRST' | 'AFTER' ColumnName)?

11.5.2.61.2 Examples

Meta-Only Change

```
CREATE TABLE t1 (id int not null primary key AUTO_INCREMENT, col1 INT);
```

```
Query OK, 0 rows affected (0.11 sec)
```

```
INSERT INTO t1 (col1) VALUES (1),(2),(3),(4),(5);
```

```
Query OK, 5 rows affected (0.02 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

```
ALTER TABLE t1 MODIFY col1 BIGINT;
```

```
Query OK, 0 rows affected (0.09 sec)
```

```
SHOW CREATE TABLE t1\G;
```

```
***** 1. row *****
Table: t1
Create Table: CREATE TABLE `t1` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `col1` bigint(20) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin AUTO_INCREMENT
→ =30001
1 row in set (0.00 sec)
```

Reorg-Data Change

```
CREATE TABLE t1 (id int not null primary key AUTO_INCREMENT, col1 INT);
```

```
Query OK, 0 rows affected (0.11 sec)
```

```
INSERT INTO t1 (col1) VALUES (12345),(67890);
```

```
Query OK, 2 rows affected (0.00 sec)
Records: 2 Duplicates: 0 Warnings: 0
```

```
ALTER TABLE t1 MODIFY col1 VARCHAR(5);
```

```
Query OK, 0 rows affected (2.52 sec)
```

```
SHOW CREATE TABLE t1\G;
```

```
***** 1. row *****
Table: t1
CREATE TABLE `t1` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `col1` varchar(5) DEFAULT NULL,
  PRIMARY KEY (`id`) /*T![clustered_index] CLUSTERED */
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin AUTO_INCREMENT
→ =30001
1 row in set (0.00 sec)
```

Note:

- TiDB returns an error when the changed data type conflicts with an existing data row. In the above example, TiDB returns the following error:

```
alter table t1 modify column col1 varchar(4); ERROR 1406
→ (22001): Data Too Long, field len 4, data len 5
```

- Due to the compatibility with the Async Commit feature, the DDL statement waits for a period of time (about 2.5s) before starting to process into Reorg Data.

```
Query OK, 0 rows affected (2.52 sec)
```

11.5.2.61.3 MySQL compatibility

- Does not support modifying multiple columns using a single ALTER TABLE statement. For example:

```
ALTER TABLE t1 MODIFY col1 BIGINT, MODIFY id BIGINT NOT NULL;
ERROR 1105 (HY000): Unsupported multi schema change
```

- Does not support modifying the Reorg-Data types on the primary key columns but supports modifying the Meta-Only types. For example:

```
CREATE TABLE t (a int primary key);
ALTER TABLE t MODIFY COLUMN a VARCHAR(10);
ERROR 8200 (HY000): Unsupported modify column: column has primary key
→ flag
```

```
CREATE TABLE t (a int primary key);
ALTER TABLE t MODIFY COLUMN a bigint;
Query OK, 0 rows affected (0.01 sec)
```

- Does not support modifying the column types on generated columns. For example:

```
CREATE TABLE t (a INT, b INT as (a+1));
ALTER TABLE t MODIFY COLUMN b VARCHAR(10);
ERROR 8200 (HY000): Unsupported modify column: column is generated
```

- Does not support modifying the column types on the partitioned tables. For example:

```

CREATE TABLE t (c1 INT, c2 INT, c3 INT) partition by range columns(c1)
    ↪ ( partition p0 values less than (10), partition p1 values less
    ↪ than (maxvalue));
ALTER TABLE t MODIFY COLUMN c1 DATETIME;
ERROR 8200 (HY000): Unsupported modify column: table is partition
    ↪ table

```

- Does not support modifying some data types (for example, some TIME types, Bit, Set, Enum, JSON) are not supported due to some compatibility issues of the `cast` function's behavior between TiDB and MySQL.

```

CREATE TABLE t (a DECIMAL(13, 7));
ALTER TABLE t MODIFY COLUMN a DATETIME;
ERROR 8200 (HY000): Unsupported modify column: change from original
    ↪ type decimal(13,7) to datetime is currently unsupported yet

```

11.5.2.61.4 See also

- [CREATE TABLE](#)
- [SHOW CREATE TABLE](#)
- [ADD COLUMN](#)
- [DROP COLUMN](#)
- [CHANGE COLUMN](#)

11.5.2.62 PREPARE

The `PREPARE` statement provides an SQL interface to server-side prepared statements.

11.5.2.62.1 Synopsis

```

PreparedStmt ::= 
    'PREPARE' Identifier 'FROM' PrepareSQL

PrepareSQL ::= 
    stringLit
    | UserVariable

```

11.5.2.62.2 Examples

```

mysql> PREPARE mystmt FROM 'SELECT ? as num FROM DUAL';
Query OK, 0 rows affected (0.00 sec)

mysql> SET @number = 5;

```

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

```
mysql> EXECUTE mystmt USING @number;
+-----+
| num |
+-----+
| 5   |
+-----+
1 row in set (0.00 sec)
```

```
mysql> DEALLOCATE PREPARE mystmt;
Query OK, 0 rows affected (0.00 sec)
```

11.5.2.62.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be [reported via an issue](#) on GitHub.

11.5.2.62.4 See also

- [EXECUTE](#)
- [DEALLOCATE](#)

11.5.2.63 RECOVER TABLE

`RECOVER TABLE` is used to recover a deleted table and the data on it within the GC (Garbage Collection) life time after the `DROP TABLE` statement is executed.

11.5.2.63.1 Syntax

```
RECOVER TABLE table_name
```

```
RECOVER TABLE BY JOB ddl_job_id
```

11.5.2.63.2 Synopsis

```
RecoverTableStmt ::=  
    'RECOVER' 'TABLE' ( 'BY' 'JOB' Int64Num | TableName Int64Num? )  
  
TableName ::=  
    Identifier ( '.' Identifier )?  
  
Int64Num ::= NUM  
  
NUM ::= intLit
```

Note:

- If a table is deleted and the GC lifetime is out, the table cannot be recovered with `RECOVER TABLE`. Execution of `RECOVER TABLE` in this scenario returns an error like: `snapshot is older than GC safe point ↵ 2019-07-10 13:45:57 +0800 CST.`
- If the TiDB version is 3.0.0 or later, it is not recommended for you to use `RECOVER TABLE` when TiDB Binlog is used.
- `RECOVER TABLE` is supported in the Binlog version 3.0.1, so you can use `RECOVER TABLE` in the following three situations:
 - Binlog version is 3.0.1 or later.
 - TiDB 3.0 is used both in the upstream cluster and the downstream cluster.
 - The GC life time of the secondary cluster must be longer than that of the primary cluster. However, as latency occurs during data replication between upstream and downstream databases, data recovery might fail in the downstream.

Troubleshoot errors during TiDB Binlog replication

When you use `RECOVER TABLE` in the upstream TiDB during TiDB Binlog replication, TiDB Binlog might be interrupted in the following three situations:

- The downstream database does not support the `RECOVER TABLE` statement. An error instance: `check the manual that corresponds to your MySQL server version ↵ for the right syntax to use near 'RECOVER TABLE table_name'.`
- The GC life time is not consistent between the upstream database and the downstream database. An error instance: `snapshot is older than GC safe point 2019-07-10 ↵ 13:45:57 +0800 CST.`
- Latency occurs during replication between upstream and downstream databases. An error instance: `snapshot is older than GC safe point 2019-07-10 13:45:57 ↵ +0800 CST.`

For the above three situations, you can resume data replication from TiDB Binlog with a **full import of the deleted table**.

11.5.2.63.3 Examples

- Recover the deleted table according to the table name.

```
DROP TABLE t;
```

```
RECOVER TABLE t;
```

This method searches the recent DDL job history and locates the first DDL operation of the `DROP TABLE` type, and then recovers the deleted table with the name identical to the one table name specified in the `RECOVER TABLE` statement.

- Recover the deleted table according to the table's DDL JOB ID used.

Suppose that you had deleted the table `t` and created another `t`, and again you deleted the newly created `t`. Then, if you want to recover the `t` deleted in the first place, you must use the method that specifies the DDL JOB ID.

```
DROP TABLE t;
```

```
ADMIN SHOW DDL JOBS 1;
```

The second statement above is used to search for the table's DDL JOB ID to delete `t`. In the following example, the ID is 53.

JOB_ID	DB_NAME	TABLE_NAME	JOB_TYPE	SCHEMA_STATE	SCHEMA_ID
					STATE
53	test		drop table	none	1 41
					synced

```
RECOVER TABLE BY JOB 53;
```

This method recovers the deleted table via the DDL JOB ID. If the corresponding DDL job is not of the `DROP TABLE` type, an error occurs.

11.5.2.63.4 Implementation principle

When deleting a table, TiDB only deletes the table metadata, and writes the table data (row data and index data) to be deleted to the `mysql.gc_delete_range` table. The GC Worker in the TiDB background periodically removes from the `mysql.gc_delete_range` table the keys that exceed the GC life time.

Therefore, to recover a table, you only need to recover the table metadata and delete the corresponding row record in the `mysql.gc_delete_range` table before the GC Worker

deletes the table data. You can use a snapshot read of TiDB to recover the table metadata. Refer to [Read Historical Data](#) for details.

Table recovery is done by TiDB obtaining the table metadata through snapshot read, and then going through the process of table creation similar to `CREATE TABLE`. Therefore, `RECOVER TABLE` itself is, in essence, a kind of DDL operation.

11.5.2.63.5 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.64 RENAME INDEX

The statement `ALTER TABLE ... RENAME INDEX` renames an existing index to a new name. This operation is instant in TiDB, and requires only a meta data change.

11.5.2.64.1 Synopsis

```
AlterTableStmt ::=  
  'ALTER' IgnoreOptional 'TABLE' TableName ( AlterTableSpecListOpt  
    ↪ AlterTablePartitionOpt | 'ANALYZE' 'PARTITION' PartitionNameList (   
    ↪ 'INDEX' IndexNameList )? AnalyzeOptionListOpt )  
  
KeyOrIndex ::=  
  'KEY'  
  | 'INDEX'
```

11.5.2.64.2 Examples

```
mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT  
    ↪ NOT NULL, INDEX col1 (c1));  
Query OK, 0 rows affected (0.11 sec)  
  
mysql> SHOW CREATE TABLE t1\G  
***** 1. row *****  
      Table: t1  
Create Table: CREATE TABLE `t1` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `c1` int(11) NOT NULL,  
  PRIMARY KEY (`id`),  
  KEY `col1` (`c1`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin  
1 row in set (0.00 sec)  
  
mysql> ALTER TABLE t1 RENAME INDEX col1 TO c1;  
Query OK, 0 rows affected (0.09 sec)
```

```
mysql> SHOW CREATE TABLE t1\G
***** 1. row *****
      Table: t1
Create Table: CREATE TABLE `t1` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `c1` int(11) NOT NULL,
  PRIMARY KEY (`id`),
  KEY `c1` (`c1`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin
1 row in set (0.00 sec)
```

11.5.2.64.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.64.4 See also

- [SHOW CREATE TABLE](#)
- [CREATE INDEX](#)
- [DROP INDEX](#)
- [SHOW INDEX](#)
- [ALTER INDEX](#)

11.5.2.65 RENAME TABLE

This statement renames an existing table to a new name.

11.5.2.65.1 Synopsis

```
RenameTableStmt ::=

  'RENAME' 'TABLE' TableToTable ( ',' TableToTable )*

TableToTable ::=

  TableName 'TO' TableName
```

11.5.2.65.2 Examples

```
mysql> CREATE TABLE t1 (a int);
Query OK, 0 rows affected (0.12 sec)

mysql> SHOW TABLES;
+-----+
```

```

| Tables_in_test |
+-----+
| t1      |
+-----+
1 row in set (0.00 sec)

mysql> RENAME TABLE t1 TO t2;
Query OK, 0 rows affected (0.08 sec)

mysql> SHOW TABLES;
+-----+
| Tables_in_test |
+-----+
| t2      |
+-----+
1 row in set (0.00 sec)

```

11.5.2.65.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.65.4 See also

- [CREATE TABLE](#)
- [SHOW TABLES](#)
- [ALTER TABLE](#)

11.5.2.66 REPLACE

The REPLACE statement is semantically a combined DELETE+INSERT statement. It can be used to simplify application code.

11.5.2.66.1 Synopsis

```

ReplaceIntoStmt ::=

  'REPLACE' PriorityOpt IntoOpt TableName PartitionNameListOpt
    ↪ InsertValues

PriorityOpt ::=
  ( 'LOW_PRIORITY' | 'HIGH_PRIORITY' | 'DELAYED' )?

IntoOpt ::= 'INTO'?

```

```

TableName ::= Identifier ('.' Identifier )?

PartitionNameListOpt ::= ('PARTITION' '(' Identifier (',' Identifier )* ')')?

InsertValues ::= '(' (ColumnNameListOpt) (ValueSym ValuesList | SelectStmt | '('
    → SelectStmt ')' | UnionStmt ) | SelectStmt ')'
| ValueSym ValuesList
| SelectStmt
| UnionStmt
| 'SET' ColumnSetValue? (',' ColumnSetValue)*

```

11.5.2.66.2 Examples

```

mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT
    → NOT NULL);
Query OK, 0 rows affected (0.12 sec)

mysql> INSERT INTO t1 (c1) VALUES (1), (2), (3);
Query OK, 3 rows affected (0.02 sec)
Records: 3 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM t1;
+---+---+
| id | c1 |
+---+---+
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
+---+---+
3 rows in set (0.00 sec)

mysql> REPLACE INTO t1 (id, c1) VALUES(3, 99);
Query OK, 2 rows affected (0.01 sec)

mysql> SELECT * FROM t1;
+---+---+
| id | c1 |
+---+---+
| 1 | 1 |
| 2 | 2 |
| 3 | 99 |
+---+---+

```

```
+-----+
3 rows in set (0.00 sec)
```

11.5.2.66.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.66.4 See also

- [DELETE](#)
- [INSERT](#)
- [SELECT](#)
- [UPDATE](#)

11.5.2.67 RESTORE

This statement performs a distributed restore from a backup archive previously produced by a [BACKUP statement](#).

The `RESTORE` statement uses the same engine as the [BR tool](#), except that the restore process is driven by TiDB itself rather than a separate BR tool. All benefits and caveats of BR also apply here. In particular, **RESTORE is currently not ACID-compliant**. Before running `RESTORE`, ensure that the following requirements are met:

- The cluster is “offline”, and the current TiDB session is the only active SQL connection to access all tables being restored.
- When a full restore is being performed, the tables being restored should not already exist, because existing data might be overridden and causes inconsistency between the data and indices.
- When an incremental restore is being performed, the tables should be at the exact same state as the `LAST_BACKUP` timestamp when the backup is created.

Running `RESTORE` requires either the `RESTORE_ADMIN` or `SUPER` privilege. Additionally, both the TiDB node executing the restore and all TiKV nodes in the cluster must have read permission from the destination.

The `RESTORE` statement is blocking, and will finish only after the entire restore task is finished, failed, or canceled. A long-lasting connection should be prepared for running `RESTORE`. The task can be canceled using the [KILL TIDB QUERY](#) statement.

Only one `BACKUP` and `RESTORE` task can be executed at a time. If a `BACKUP` or `RESTORE` task is already running on the same TiDB server, the new `RESTORE` execution will wait until all previous tasks are done.

`RESTORE` can only be used with “`tikv`” storage engine. Using `RESTORE` with the “`unistore`” engine will fail.

11.5.2.67.1 Synopsis

```

RestoreStmt ::= "RESTORE" BRIETables "FROM" stringLit RestoreOption*
BRIETables ::= "DATABASE" ('*' | DBName (',' DBName)* )
| "TABLE" TableNameList
RestoreOption ::= "RATE_LIMIT" '='? LengthNum "MB" '/' "SECOND"
| "CONCURRENCY" '='? LengthNum
| "CHECKSUM" '='? Boolean
| "SEND_CREDENTIALS_TO_TIKV" '='? Boolean
Boolean ::= NUM | "TRUE" | "FALSE"

```

11.5.2.67.2 Examples

Restore from backup archive

```
RESTORE DATABASE * FROM 'local:///mnt/backup/2020/04/';
```

Destination	Size	BackupTS	Queue Time
Execution Time			
local:///mnt/backup/2020/04/	248665063	0	2020-04-21 17:16:55
			2020-04-21 17:16:55

1 row in set (28.961 sec)

In the example above, all data is restored from a backup archive at the local filesystem. The data is read as SST files from the /mnt/backup/2020/04/ directories distributed among all TiDB and TiKV nodes.

The first row of the result above is described as follows:

Column	Description
Destination	The → destination URL to read from
Size	The total size of the backup archive, in bytes
BackupTS	(not used)
Queue	The → Time timestamp (in current time zone) when the RESTORE task was queued.
Execution	The → Time timestamp (in current time zone) when the RESTORE task starts to run.

Partial restore

You can specify which databases or tables to restore. If some databases or tables are missing from the backup archive, they will be ignored, and thus RESTORE would complete without doing anything.

```
RESTORE DATABASE `test` FROM 'local:///mnt/backup/2020/04/' ;
```

```
RESTORE TABLE `test`.`sbtest01`, `test`.`sbtest02` FROM 'local:///mnt/
↪ backup/2020/04/' ;
```

External storages

BR supports restoring data from S3 or GCS:

```
RESTORE DATABASE * FROM 's3://example-bucket-2020/backup-05/?region=us-west
↪ -2' ;
```

The URL syntax is further explained in [External Storages](#).

When running on cloud environment where credentials should not be distributed, set the `SEND_CREDENTIALS_TO_TIKV` option to `FALSE`:

```
RESTORE DATABASE * FROM 's3://example-bucket-2020/backup-05/?region=us-west
↪ -2'
SEND_CREDENTIALS_TO_TIKV = FALSE;
```

Performance fine-tuning

Use `RATE_LIMIT` to limit the average download speed per TiKV node to reduce network bandwidth.

By default, TiDB node would run 128 restore threads. This value can be adjusted with the `CONCURRENCY` option.

Before restore is completed, `RESTORE` would perform a checksum against the data from the archive to verify correctness. This step can be disabled with the `CHECKSUM` option if you are confident that this is unnecessary.

```
RESTORE DATABASE * FROM 's3://example-bucket-2020/backup-06/'
RATE_LIMIT = 120 MB/SECOND
CONCURRENCY = 64
CHECKSUM = FALSE;
```

Incremental restore

There is no special syntax to perform incremental restore. TiDB will recognize whether the backup archive is full or incremental and take appropriate action. You only need to apply each incremental restore in correct order.

For instance, if a backup task is created as follows:

```
BACKUP DATABASE `test` TO 's3://example-bucket/full-backup' SNAPSHOT =
↪ 413612900352000;
BACKUP DATABASE `test` TO 's3://example-bucket/inc-backup-1' SNAPSHOT =
↪ 414971854848000 LAST_BACKUP = 413612900352000;
BACKUP DATABASE `test` TO 's3://example-bucket/inc-backup-2' SNAPSHOT =
↪ 416353458585600 LAST_BACKUP = 414971854848000;
```

then the same order should be applied in the restore:

```
RESTORE DATABASE * FROM 's3://example-bucket/full-backup';
RESTORE DATABASE * FROM 's3://example-bucket/inc-backup-1';
RESTORE DATABASE * FROM 's3://example-bucket/inc-backup-2';
```

11.5.2.67.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.67.4 See also

- [BACKUP](#)
- [SHOW RESTORES](#)

11.5.2.68 REVOKE <privileges>

This statement removes privileges from an existing user. Executing this statement requires the GRANT OPTION privilege and all privileges you revoke.

11.5.2.68.1 Synopsis

```

GrantStmt ::=

  'GRANT' PrivElemList 'ON' ObjectType PrivLevel 'TO' UserSpecList
    ↪ RequireClauseOpt WithGrantOptionOpt

PrivElemList ::=

  PrivElem ( ',' PrivElem )*

PrivElem ::=

  PrivType ( '(' ColumnNameList ')' )?

PrivType ::=

  'ALL' 'PRIVILEGES'?
  | 'ALTER' 'ROUTINE'?
  | 'CREATE' ( 'USER' | 'TEMPORARY' 'TABLES' | 'VIEW' | 'ROLE' | 'ROUTINE' )
    ↪ ?
  | 'TRIGGER'
  | 'DELETE'
  | 'DROP' 'ROLE'?
  | 'PROCESS'
  | 'EXECUTE'
  | 'INDEX'
  | 'INSERT'
  | 'SELECT'
  | 'SUPER'
  | 'SHOW' ( 'DATABASES' | 'VIEW' )
  | 'UPDATE'
  | 'GRANT' 'OPTION'
  | 'REFERENCES'
  | 'REPLICATION' ( 'SLAVE' | 'CLIENT' )
  | 'USAGE'
  | 'RELOAD'
  | 'FILE'
  | 'CONFIG'

```

```

|   'LOCK' 'TABLES'
|   'EVENT'
|   'SHUTDOWN'

ObjectType ::=

    'TABLE'?

PrivLevel ::=

    '*' ('.' '*' )?
| Identifier ('.' ('*' | Identifier ) )?

UserSpecList ::=

    UserSpec ( ',' UserSpec )*

```

11.5.2.68.2 Examples

```
mysql> CREATE USER 'newuser' IDENTIFIED BY 'mypassword';
Query OK, 1 row affected (0.02 sec)
```

```
mysql> GRANT ALL ON test.* TO 'newuser';
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> SHOW GRANTS FOR 'newuser';
+-----+
| Grants for newuser@%          |
+-----+
| GRANT USAGE ON *.* TO 'newuser'@'%'
| GRANT ALL PRIVILEGES ON test.* TO 'newuser'@'%'
+-----+
2 rows in set (0.00 sec)
```

```
mysql> REVOKE ALL ON test.* FROM 'newuser';
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> SHOW GRANTS FOR 'newuser';
+-----+
| Grants for newuser@%          |
+-----+
| GRANT USAGE ON *.* TO 'newuser'@'%'
+-----+
1 row in set (0.00 sec)
```

```
mysql> DROP USER 'newuser';
Query OK, 0 rows affected (0.14 sec)
```

```
mysql> SHOW GRANTS FOR 'newuser';
ERROR 1141 (42000): There is no such grant defined for user 'newuser' on
↪ host '%'
```

11.5.2.68.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.68.4 See also

- [GRANT <privileges>](#)
- [SHOW GRANTS](#)
- [Privilege Management](#)

11.5.2.69 REVOKE <role>

This statement removes a previously assigned role from a specified user (or list of users).

11.5.2.69.1 Synopsis

```
RevokeRoleStmt ::=

    'REVOKE' RolenameList 'FROM' UsernameList

RolenameList ::=

    Rolename ( ',' Rolename )*

UsernameList ::=

    Username ( ',' Username )*
```

11.5.2.69.2 Examples

Create a new role for the analytics team, and a new user called jennifer:

```
$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 37
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
↪ License 2.0) Community Edition, MySQL 5.7 compatible

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> CREATE ROLE analyticsteam;
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> GRANT SELECT ON test.* TO analyticsteam;
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> CREATE USER jennifer;
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> GRANT analyticsteam TO jennifer;
Query OK, 0 rows affected (0.01 sec)
```

Note that by default **jennifer** needs to SET ROLE **analyticsteam** in order to be able to use the privileges associated with the role:

```
$ mysql -ujennifer
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 32
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible
```

```
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> SHOW GRANTS;
+-----+
| Grants for User           |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
```

```

2 rows in set (0.00 sec)

mysql> SHOW TABLES in test;
ERROR 1044 (42000): Access denied for user 'jennifer'@'%' to database 'test'
mysql> SET ROLE analyticsteam;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT Select ON test.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
3 rows in set (0.00 sec)

mysql> SHOW TABLES IN test;
+-----+
| Tables_in_test |
+-----+
| t1           |
+-----+
1 row in set (0.00 sec)

```

The statement `SET DEFAULT ROLE` can be used to associate a role to `jennifer` so that she will not have to execute the statement `SET ROLE` in order to assume the privileges associated with the role:

```

$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 34
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible

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> SET DEFAULT ROLE analyticsteam TO jennifer;
Query OK, 0 rows affected (0.02 sec)
```

```
$ mysql -ujennifer
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 35
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible

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> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT Select ON test.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
3 rows in set (0.00 sec)

mysql> SHOW TABLES IN test;
+-----+
| Tables_in_test |
+-----+
| t1           |
+-----+
1 row in set (0.00 sec)
```

Revoke the role of analyticsteam from jennifer:

```
$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 38
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible
```

```
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> REVOKE analyticsteam FROM jennifer;
Query OK, 0 rows affected (0.01 sec)
```

```
$ mysql -ujennifer
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 39
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible
```

```
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> SHOW GRANTS;
+-----+
| Grants for User           |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
+-----+
1 row in set (0.00 sec)
```

11.5.2.69.3 MySQL compatibility

This statement is understood to be fully compatible with roles, which are a feature of MySQL 8.0. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.69.4 See also

- `CREATE ROLE`
- `DROP ROLE`
- `GRANT <role>`
- `SET ROLE`
- `SET DEFAULT ROLE`
- Role-Based Access Control

11.5.2.70 ROLLBACK

This statement reverts all changes in the current transaction inside of TiDB. It is the opposite of a `COMMIT` statement.

11.5.2.70.1 Synopsis

```
RollbackStmt ::=  
    'ROLLBACK' CompletionTypeWithinTransaction?  
  
CompletionTypeWithinTransaction ::=  
    'AND' ( 'CHAIN' ( 'NO' 'RELEASE' )? | 'NO' 'CHAIN' ( 'NO'? 'RELEASE' )?  
          ↪ )  
    | 'NO'? 'RELEASE'
```

11.5.2.70.2 Examples

```
mysql> CREATE TABLE t1 (a INT NOT NULL PRIMARY KEY);  
Query OK, 0 rows affected (0.12 sec)  
  
mysql> BEGIN;  
Query OK, 0 rows affected (0.00 sec)  
  
mysql> INSERT INTO t1 VALUES (1);  
Query OK, 1 row affected (0.00 sec)  
  
mysql> ROLLBACK;  
Query OK, 0 rows affected (0.01 sec)  
  
mysql> SELECT * FROM t1;  
Empty set (0.01 sec)
```

11.5.2.70.3 MySQL compatibility

- TiDB does not support savepoints or the syntax `ROLLBACK TO SAVEPOINT`.

- TiDB parses but ignores the syntax ROLLBACK AND [NO] RELEASE. This functionality is used in MySQL to disconnect the client session immediately after rolling back the transaction. In TiDB, it is recommended to instead use the `mysql_close()` functionality of your client driver.
- TiDB parses but ignores the syntax ROLLBACK AND [NO] CHAIN. This functionality is used in MySQL to immediately start a new transaction with the same isolation level while the current transaction is being rolled back. In TiDB, it is recommended to instead start a new transaction.

11.5.2.70.4 See also

- [COMMIT](#)
- [BEGIN](#)
- [START TRANSACTION](#)

11.5.2.71 SELECT

The SELECT statement is used to read data from TiDB.

11.5.2.71.1 Synopsis

SelectStmt:

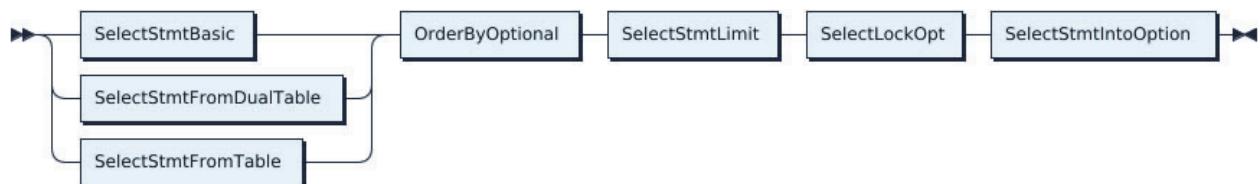


Figure 202: SelectStmt

FromDual:



Figure 203: FromDual

WhereClauseOptional:

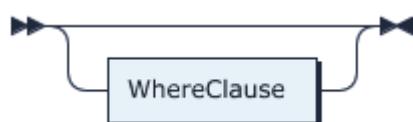


Figure 204: WhereClauseOptional

SelectStmtOpts:

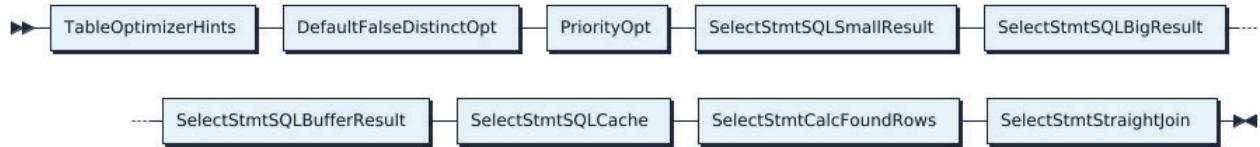


Figure 205: SelectStmtOpts

SelectStmtFieldList:

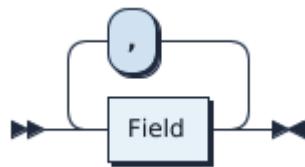


Figure 206: SelectStmtFieldList

TableRefsClause:

```

TableRefsClause ::= 
  TableRef AsOfClause? ( ',' TableRef AsOfClause? )*
AsOfClause ::= 
  'AS' 'OF' 'TIMESTAMP' Expression
  
```

WhereClauseOptional:

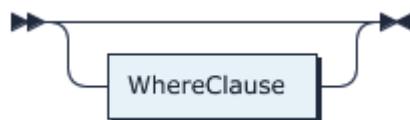


Figure 207: WhereClauseOptional

SelectStmtGroup:

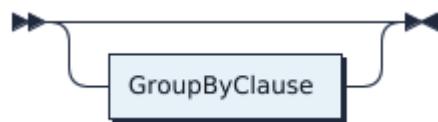


Figure 208: SelectStmtGroup

HavingClause:

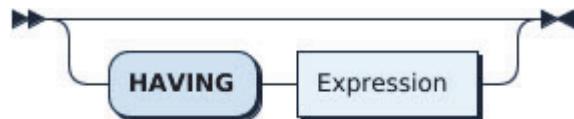


Figure 209: HavingClause

`OrderByOptional`:

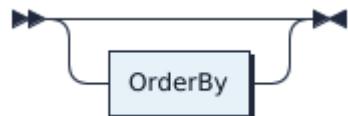


Figure 210: OrderByOptional

`SelectStmtLimit`:

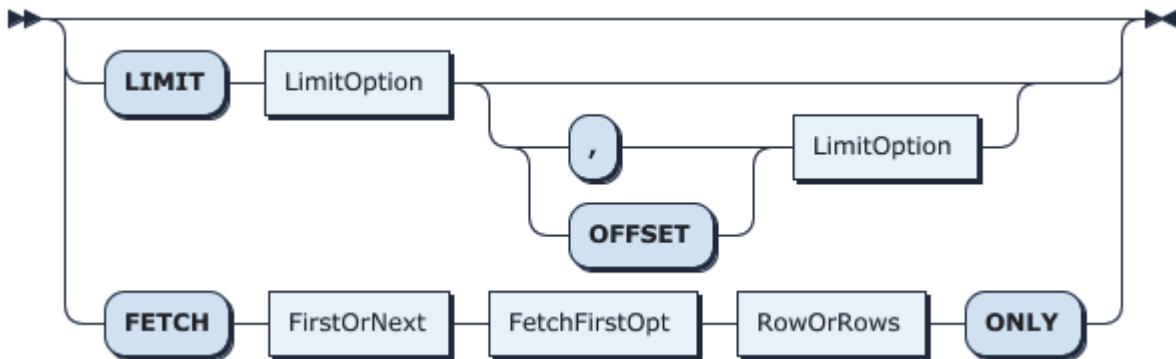


Figure 211: SelectStmtLimit

`FirstOrNext`:

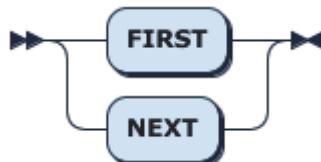


Figure 212: FirstOrNext

`FetchFirstOpt`:



Figure 213: FetchFirstOpt

RowOrRows:

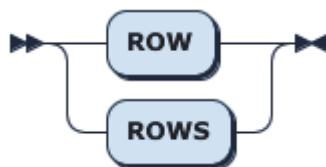


Figure 214: RowOrRows

SelectLockOpt:

```

SelectLockOpt ::= 
  ( ( 'FOR' 'UPDATE' ( 'OF' TableList )? 'NOWAIT'? )
  | ( 'LOCK' 'IN' 'SHARE' 'MODE' )? )

TableList ::= 
  TableName ( ',' TableName )*

```

WindowClauseOptional

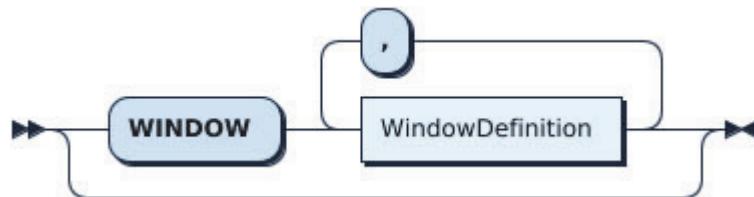


Figure 215: WindowClauseOptional

11.5.2.71.2 Description of the syntax elements

Syntax Element	Description
TableOptimizerHints	This is the hint to control the behavior of TiDB's optimizer. For more information, refer to Optimizer Hints .
ALL, DISTINCT, DISTINCTROW	The ALL, DISTINCT/DISTINCTROW modifiers specify whether duplicate rows should be returned. ALL (the default) specifies that all matching rows should be returned.
HIGH_PRIORITY	HIGH_PRIORITY gives the current statement higher priority than other statements.
SQL_CALC_FOUND_ROWS	TiDB does not support this feature, and will return an error unless <code>tidb_enable_noop_functions=1</code> is set.

Syntax Element	Description
SQL_CACHE, SQL_NO_CACHE	SQL_CACHE and SQL_NO_CACHE are used to control whether to cache the request results to the BlockCache of TiKV (RocksDB). For a one-time query on a large amount of data, such as the count(*) query, it is recommended to fill in SQL_NO_CACHE to avoid flushing the hot user data in BlockCache.
STRAIGHT_JOIN	STRAIGHT_JOIN forces the optimizer to do a union query in the order of the tables used in the FROM clause. When the optimizer chooses a join order that is not good, you can use this syntax to speed up the execution of the query.
select_expr	Each select_expr indicates a column to retrieve. including the column names and expressions. * represents all the columns.
FROM ↪ table_references	The FROM table_references clause indicates the table (such as select * from t;), or tables (such as select * from t1 join t2;) or even 0 tables (such as select 1+1 from dual; which is equivalent to select 1+1;) from which to retrieve rows.
WHERE ↪ where_condition	The WHERE clause, if given, indicates the condition or conditions that rows must satisfy to be selected. The result contains only the data that meets the condition(s).
GROUP BY HAVING ↪ where_condition	The GROUP BY statement is used to group the result-set. The HAVING clause and the WHERE clause are both used to filter the results. The HAVING clause filters the results of GROUP BY, while the WHERE clause filter the results before aggregation.
ORDER BY	The ORDER BY clause is used to sort the data in ascending or descending order, based on columns, expressions or items in the select_expr list.
LIMIT	The LIMIT clause can be used to constrain the number of rows. LIMIT takes one or two numeric arguments. With one argument, the argument specifies the maximum number of rows to return, the first row to return is the first row of the table by default; with two arguments, the first argument specifies the offset of the first row to return, and the second specifies the maximum number of rows to return. TiDB also supports the FETCH FIRST/NEXT n ROW/ROWS ONLY syntax, which has the same effect as LIMIT n. You can omit n in this syntax and its effect is the same as LIMIT 1.

Syntax Element	Description
Window ↪ <code>window_definition</code>	This is the syntax for window function, which is usually used to do some analytical computation. For more information, refer to Window Function .
<code>FOR UPDATE</code>	The <code>SELECT FOR UPDATE</code> clause locks all the data in the result sets to detect concurrent updates from other transactions. Data that match the query conditions but do not exist in the result sets are not read-locked, such as the row data written by other transactions after the current transaction is started. TiDB uses the Optimistic Transaction Model . The transaction conflicts are not detected in the statement execution phase. Therefore, the current transaction does not block other transactions from executing <code>UPDATE</code> , <code>DELETE</code> or <code>SELECT FOR UPDATE</code> like other databases such as PostgreSQL. In the committing phase, the rows read by <code>SELECT FOR UPDATE</code> are committed in two phases, which means they can also join the conflict detection. If write conflicts occur, the commit fails for all transactions that include the <code>SELECT FOR UPDATE</code> clause. If no conflict is detected, the commit succeeds. And a new version is generated for the locked rows, so that write conflicts can be detected when other uncommitted transactions are being committed later. When using pessimistic transaction mode, the behavior is basically the same as other databases. Refer to Difference with MySQL InnoDB to see the details. TiDB supports the <code>NOWAIT</code> modifier for <code>FOR UPDATE</code> . See TiDB Pessimistic Transaction Mode for details.
<code>LOCK IN SHARE MODE</code>	To guarantee compatibility, TiDB parses these three modifiers, but will ignore them.

11.5.2.71.3 Examples

```
mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT
   ↪ NOT NULL);
Query OK, 0 rows affected (0.11 sec)

mysql> INSERT INTO t1 (c1) VALUES (1),(2),(3),(4),(5);
Query OK, 5 rows affected (0.03 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM t1;
+-----+
```

```

| id | c1 |
+----+---+
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
+----+---+
5 rows in set (0.00 sec)

```

11.5.2.71.4 MySQL compatibility

- The syntax `SELECT ... INTO @variable` is not supported.
- The syntax `SELECT ... GROUP BY ... WITH ROLLUP` is not supported.
- The syntax `SELECT .. GROUP BY expr` does not imply `GROUP BY expr ORDER BY ↪ expr` as it does in MySQL 5.7. TiDB instead matches the behavior of MySQL 8.0 and does not imply a default order.

11.5.2.71.5 See also

- [INSERT](#)
- [DELETE](#)
- [UPDATE](#)
- [REPLACE](#)

11.5.2.72 SET DEFAULT ROLE

This statement sets a specific role to be applied to a user by default. Thus, they will automatically have the permissions associated with a role without having to execute `SET ↪ ROLE <rolename>` or `SET ROLE ALL`.

11.5.2.72.1 Synopsis

`SetDefaultRoleStmt:`



Figure 216: SetDefaultRoleStmt

`SetDefaultRoleOpt:`

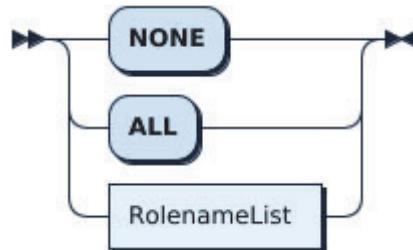


Figure 217: SetDefaultRoleOpt

RolenameList:

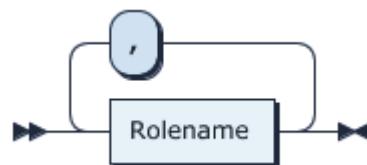


Figure 218: RolenameList

UsernameList:

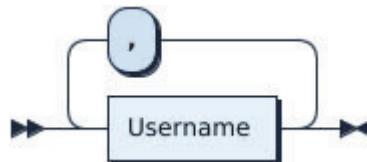


Figure 219: UsernameList

11.5.2.72.2 Examples

Create a new role for the analytics team, and a new user called jennifer:

```
$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 37
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
↳ License 2.0) Community Edition, MySQL 5.7 compatible

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> CREATE ROLE analyticsteam;
Query OK, 0 rows affected (0.02 sec)

mysql> GRANT SELECT ON test.* TO analyticsteam;
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE USER jennifer;
Query OK, 0 rows affected (0.01 sec)

mysql> GRANT analyticsteam TO jennifer;
Query OK, 0 rows affected (0.01 sec)
```

Note that by default `jennifer` needs to `SET ROLE analyticsteam` in order to be able to use the privileges associated with the role:

```
$ mysql -ujennifer
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 32
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible

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> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
2 rows in set (0.00 sec)

mysql> SHOW TABLES in test;
ERROR 1044 (42000): Access denied for user 'jennifer'@'%' to database 'test'
```

```

mysql> SET ROLE analyticsteam;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT Select ON test.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
3 rows in set (0.00 sec)

mysql> SHOW TABLES IN test;
+-----+
| Tables_in_test |
+-----+
| t1           |
+-----+
1 row in set (0.00 sec)

```

The statement `SET DEFAULT ROLE` can be used to associate a role to `jennifer` so that she will not have to execute the statement `SET ROLE` in order to assume the privileges associated with the role:

```

$ mysql -uroot
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 34
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible

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> SET DEFAULT ROLE analyticsteam TO jennifer;
Query OK, 0 rows affected (0.02 sec)

```

```
$ mysql -ujennifer
```

```
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 35
Server version: 5.7.25-TiDB-v4.0.0-beta.2-728-ga9177fe84 TiDB Server (Apache
→ License 2.0) Community Edition, MySQL 5.7 compatible

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> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT USAGE ON *.* TO 'jennifer'@'%' |
| GRANT Select ON test.* TO 'jennifer'@'%' |
| GRANT 'analyticsteam'@'%' TO 'jennifer'@'%' |
+-----+
3 rows in set (0.00 sec)

mysql> SHOW TABLES IN test;
+-----+
| Tables_in_test |
+-----+
| t1           |
+-----+
1 row in set (0.00 sec)
```

SET DEFAULT ROLE will not automatically GRANT the associated role to the user. Attempting to SET DEFAULT ROLE for a role that `jennifer` does not have granted results in the following error:

```
mysql> SET DEFAULT ROLE analyticsteam TO jennifer;
ERROR 3530 (HY000): `analyticsteam`@`%` is is not granted to jennifer@%
```

11.5.2.72.3 MySQL compatibility

This statement is understood to be fully compatible with roles, which are a feature of MySQL 8.0. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.72.4 See also

- [CREATE ROLE](#)
- [DROP ROLE](#)
- [GRANT <role>](#)
- [REVOKE <role>](#)
- [SET ROLE](#)
- [Role-Based Access Control](#)

11.5.2.73 SET [NAMES|CHARACTER SET]

The statements `SET NAMES`, `SET CHARACTER SET` and `SET CHARSET` modify the variables `character_set_client`, `character_set_results` and `character_set_connection` for the current connection.

11.5.2.73.1 Synopsis

`SetNamesStmt:`



Figure 220: `SetNamesStmt`

`VariableAssignmentList:`

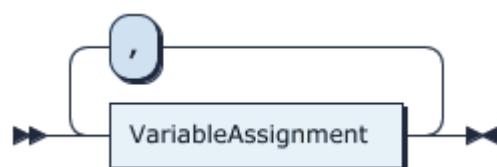


Figure 221: `VariableAssignmentList`

`VariableAssignment:`

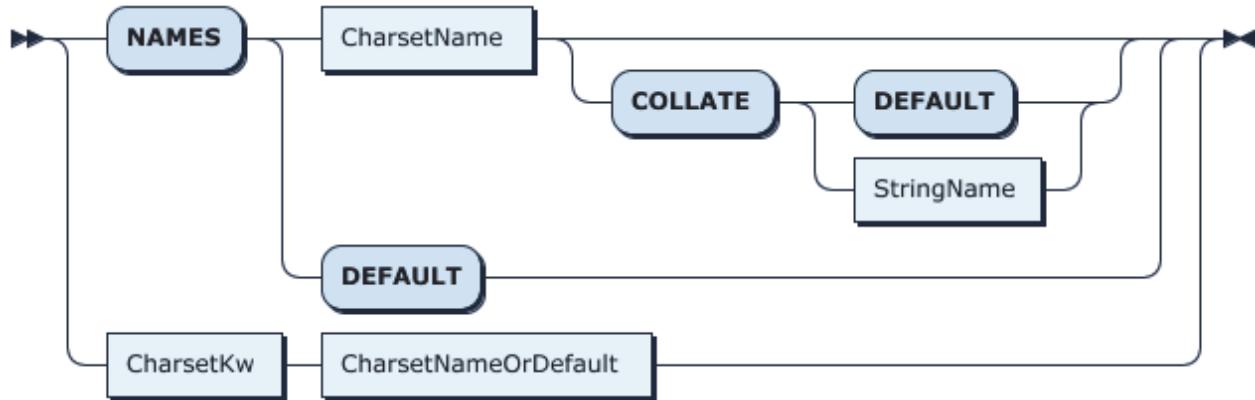


Figure 222: VariableAssignment

CharsetName:



Figure 223: CharsetName

StringName:



Figure 224: StringName

CharsetKw:

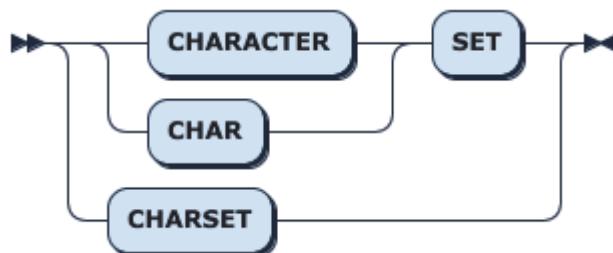


Figure 225: CharsetKw

CharsetNameOrDefault:

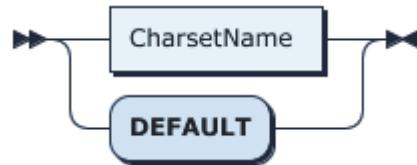


Figure 226: CharsetNameOrDefault

11.5.2.73.2 Examples

```

mysql> SHOW VARIABLES LIKE 'character_set%';
+-----+-----+
| Variable_name      | Value |
+-----+-----+
| character_sets_dir | /usr/local/mysql-5.6.25-osx10.8-x86_64/share/
|                   | charsets/ |
| character_set_connection | utf8mb4
| character_set_system | utf8
| character_set_results | utf8mb4
| character_set_client | utf8mb4
| character_set_database | utf8mb4
| character_set_filesystem | binary
| character_set_server | utf8mb4
+-----+
| Variable_name      | Value |
+-----+-----+
| character_sets_dir | /usr/local/mysql-5.6.25-osx10.8-x86_64/share/
|                   | charsets/ |
| character_set_connection | utf8
Query OK, 0 rows affected (0.00 sec)

mysql> SET NAMES utf8;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW VARIABLES LIKE 'character_set%';
+-----+-----+
| Variable_name      | Value |
+-----+-----+
| character_sets_dir | /usr/local/mysql-5.6.25-osx10.8-x86_64/share/
|                   | charsets/ |
| character_set_connection | utf8

```

```

| character_set_system | utf8
| character_set_results | utf8
| character_set_client | utf8
| character_set_server | utf8mb4
| character_set_database | utf8mb4
| character_set_filesystem | binary
+--+
    ↵ -----
    ↵
8 rows in set (0.00 sec)

mysql> SET CHARACTER SET utf8mb4;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW VARIABLES LIKE 'character_set%';
+--+
    ↵ -----
    ↵
| Variable_name          | Value
+--+
    ↵ -----
    ↵
| character_set_connection | utf8mb4
| character_set_system | utf8
| character_set_results | utf8mb4
| character_set_client | utf8mb4
| character_sets_dir     | /usr/local/mysql-5.6.25-osx10.8-x86_64/share/
    ↵ charsets/
| character_set_database | utf8mb4
| character_set_filesystem | binary
| character_set_server | utf8mb4
+--+
    ↵ -----
    ↵
8 rows in set (0.00 sec)

```

11.5.2.73.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be [reported via an issue on GitHub](#).

11.5.2.73.4 See also

- [SHOW \[GLOBAL|SESSION\] VARIABLES](#)

- `SET <variable>`
- Character Set and Collation Support

11.5.2.74 SET PASSWORD

This statement changes the user password for a user account in the TiDB system database.

11.5.2.74.1 Synopsis

`SetStmt:`

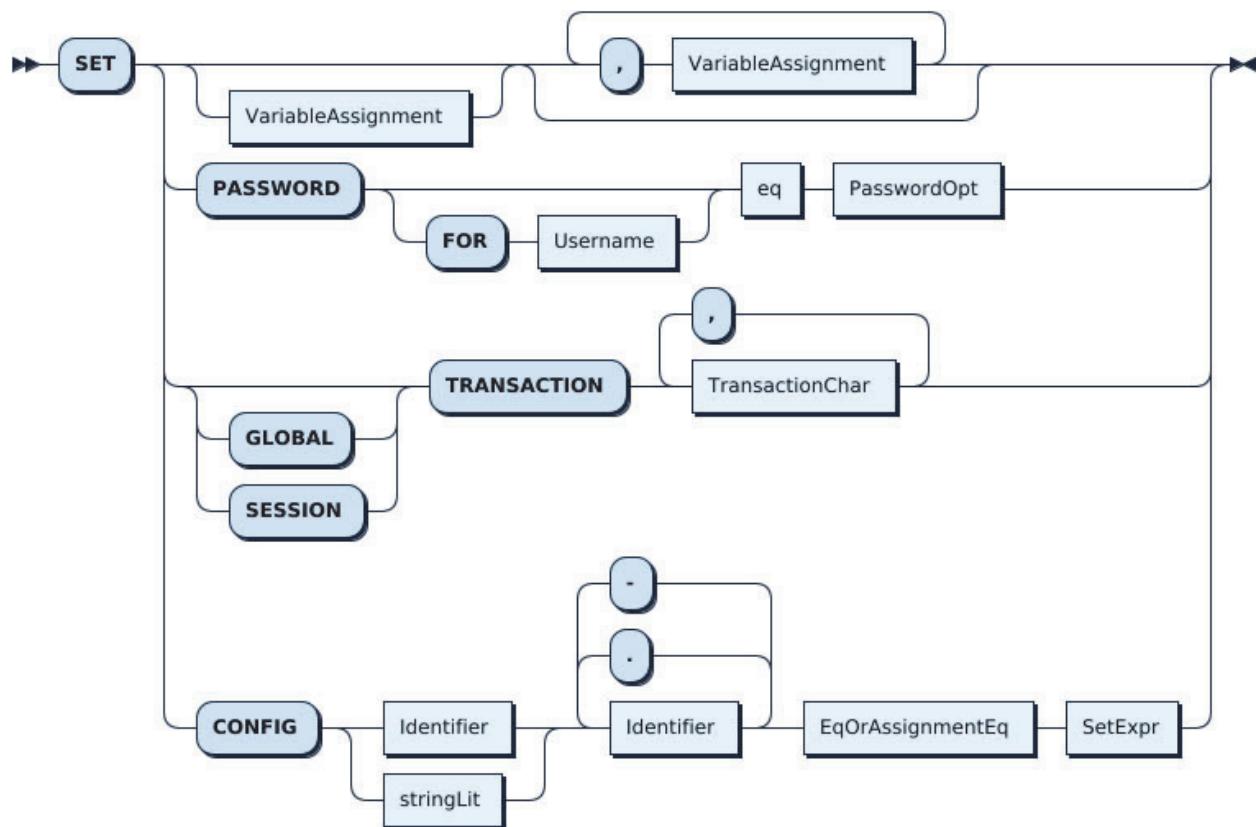


Figure 227: SetStmt

11.5.2.74.2 Examples

```
mysql> SET PASSWORD='test'; -- change my password
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> CREATE USER 'newuser' IDENTIFIED BY 'test';
Query OK, 1 row affected (0.00 sec)
```

```

mysql> SHOW CREATE USER 'newuser';
+--+
| CREATE USER for newuser@%
|   |
+--+
| CREATE USER 'newuser'@'%' IDENTIFIED WITH 'mysql_native_password' AS '*94
|   BDCEBE19083CE2A1F959FD02F964C7AF4CFC29' REQUIRE NONE PASSWORD EXPIRE
|   DEFAULT ACCOUNT UNLOCK |
+--+
|   |
+--+
1 row in set (0.00 sec)

mysql> SET PASSWORD FOR newuser = 'test';
Query OK, 0 rows affected (0.01 sec)

mysql> SHOW CREATE USER 'newuser';
+--+
| CREATE USER for newuser@%
|   |
+--+
| CREATE USER 'newuser'@'%' IDENTIFIED WITH 'mysql_native_password' AS '*94
|   BDCEBE19083CE2A1F959FD02F964C7AF4CFC29' REQUIRE NONE PASSWORD EXPIRE
|   DEFAULT ACCOUNT UNLOCK |
+--+
|   |
+--+
1 row in set (0.00 sec)

mysql> SET PASSWORD FOR newuser = PASSWORD('test'); -- deprecated syntax
        ↪ from earlier MySQL releases
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW CREATE USER 'newuser';
+--+

```

```

→
→
| CREATE USER for newuser@%
→
→ |
+-
→
→
| CREATE USER 'newuser'@'%' IDENTIFIED WITH 'mysql_native_password' AS '*94
→ BDCEBE19083CE2A1F959FD02F964C7AF4CFC29' REQUIRE NONE PASSWORD EXPIRE
→ DEFAULT ACCOUNT UNLOCK |
+-
→
→
1 row in set (0.00 sec)

```

11.5.2.74.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.74.4 See also

- [CREATE USER](#)
- [Privilege Management](#)

11.5.2.75 SET ROLE

The `SET ROLE` statement is used to enable roles in the current session. After enabling roles, users can use the privileges of the role(s).

11.5.2.75.1 Synopsis

`SetRoleStmt:`



Figure 228: SetRoleStmt

`SetRoleOpt:`

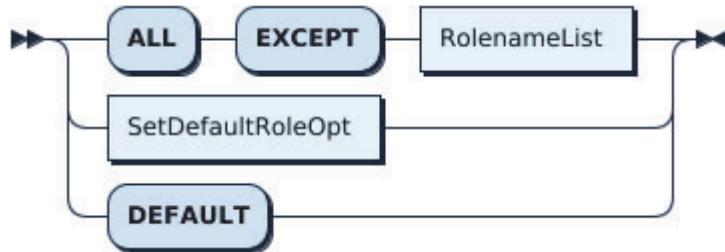


Figure 229: SetRoleOpt

SetDefaultRoleOpt:

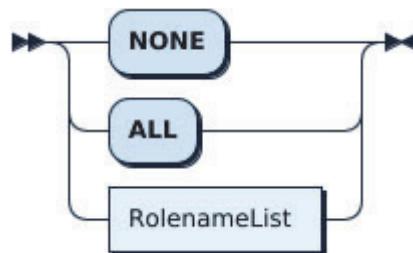


Figure 230: SetDefaultRoleOpt

11.5.2.75.2 Examples

Create a user 'u1'@'%'. and three roles: 'r1'@'%', 'r2'@'%' and 'r3'@'%'. Grant these roles to 'u1'@'%'. and set 'r1'@'%'. as the default role of 'u1'@'%'..

```

CREATE USER 'u1'@'%';
CREATE ROLE 'r1', 'r2', 'r3';
GRANT 'r1', 'r2', 'r3' TO 'u1'@'%';
SET DEFAULT ROLE 'r1' TO 'u1'@'%';

```

Log in as 'u1'@'%' and execute the following SET ROLE statement to enable all roles.

```

SET ROLE ALL;
SELECT CURRENT_ROLE();

```

CURRENT_ROLE()
`r1`@`%` , `r2`@`%` , `r3`@`%`

1 row in set (0.000 sec)

Execute the following SET ROLE statement to enable 'r2' and 'r3'.

```

SET ROLE 'r2', 'r3';
SELECT CURRENT_ROLE();

```

```
+-----+
| CURRENT_ROLE() |
+-----+
| `r2`@`%`, `r3`@`%` |
+-----+
1 row in set (0.000 sec)
```

Execute the following SET ROLE statement to enable the default role(s).

```
SET ROLE DEFAULT;
SELECT CURRENT_ROLE();
```

```
+-----+
| CURRENT_ROLE() |
+-----+
| `r1`@`%`      |
+-----+
1 row in set (0.000 sec)
```

Execute the following SET ROLE statement to cancel all enabled role(s).

```
SET ROLE NONE;
SELECT CURRENT_ROLE();
```

```
+-----+
| CURRENT_ROLE() |
+-----+
|           |
+-----+
1 row in set (0.000 sec)
```

11.5.2.75.3 MySQL compatibility

This statement is understood to be fully compatible with roles, which are a feature of MySQL 8.0. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.75.4 See also

- [CREATE ROLE](#)
- [DROP ROLE](#)
- [GRANT <role>](#)
- [REVOKE <role>](#)
- [SET DEFAULT ROLE](#)
- [Role-Based Access Control](#)

11.5.2.76 SET TRANSACTION

The `SET TRANSACTION` statement can be used to change the current isolation level on a `GLOBAL` or `SESSION` basis. This syntax is an alternative to `SET transaction_isolation='new-value'` and is included for compatibility with both MySQL, and the SQL standards.

11.5.2.76.1 Synopsis

```

SetStmt ::=

  'SET' ( VariableAssignmentList |
  'PASSWORD' ('FOR' Username)? '=' PasswordOpt |
  ( 'GLOBAL' | 'SESSION' )? 'TRANSACTION' TransactionChars |
  'CONFIG' ( Identifier | stringLit) ConfigItemName EqOrAssignmentEq
    → SetExpr )

TransactionChars ::=

  ( 'ISOLATION' 'LEVEL' IsolationLevel | 'READ' 'WRITE' | 'READ' 'ONLY'
    → AsOfClause? )

IsolationLevel ::=

  ( 'REPEATABLE' 'READ' | 'READ' ( 'COMMITTED' | 'UNCOMMITTED' ) | '
    → SERIALIZABLE' )

AsOfClause ::=

  ( 'AS' 'OF' 'TIMESTAMP' Expression)

```

11.5.2.76.2 Examples

```

mysql> SHOW SESSION VARIABLES LIKE 'transaction_isolation';
+-----+-----+
| Variable_name      | Value       |
+-----+-----+
| transaction_isolation | REPEATABLE-READ |
+-----+-----+
1 row in set (0.00 sec)

mysql> SET SESSION TRANSACTION ISOLATION LEVEL READ COMMITTED;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW SESSION VARIABLES LIKE 'transaction_isolation';
+-----+-----+
| Variable_name      | Value       |
+-----+-----+
| transaction_isolation | READ-COMMITTED |
+-----+-----+

```

```

1 row in set (0.01 sec)

mysql> SET SESSION transaction_isolation = 'REPEATABLE-READ';
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW SESSION VARIABLES LIKE 'transaction_isolation';
+-----+-----+
| Variable_name      | Value           |
+-----+-----+
| transaction_isolation | REPEATABLE-READ |
+-----+-----+
1 row in set (0.00 sec)

```

11.5.2.76.3 MySQL compatibility

- TiDB supports the ability to set a transaction as read-only in syntax only.
- The isolation levels READ-UNCOMMITTED and SERIALIZABLE are not supported.
- The REPEATABLE-READ isolation level is achieved through using the snapshot isolation technology, which is partly compatible with MySQL.
- In pessimistic transactions, TiDB supports two isolation levels compatible with MySQL: REPEATABLE-READ and READ-COMMITTED. For a detailed description, see [Isolation Levels](#).

11.5.2.76.4 See also

- [SET \[GLOBAL|SESSION\] <variable>](#)
- [Isolation Levels](#)

11.5.2.77 SET [GLOBAL|SESSION] <variable>

The statement `SET [GLOBAL|SESSION]` modifies one of TiDB's built in variables, of either SESSION or GLOBAL scope.

Note:

Similar to MySQL, changes to GLOBAL variables do not apply to either existing connections, or the local connection. Only new sessions reflect the changes to the value.

11.5.2.77.1 Synopsis

SetStmt:

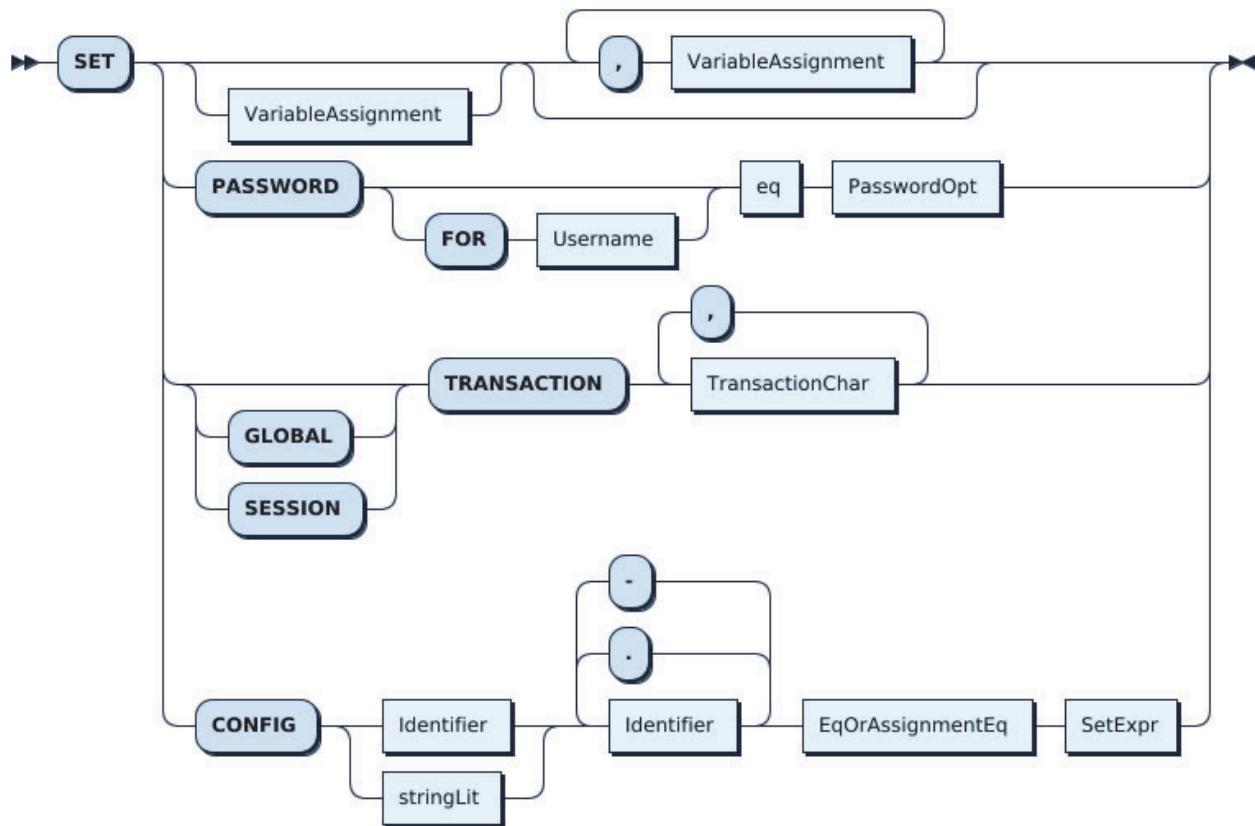


Figure 231: SetStmt

VariableAssignment:

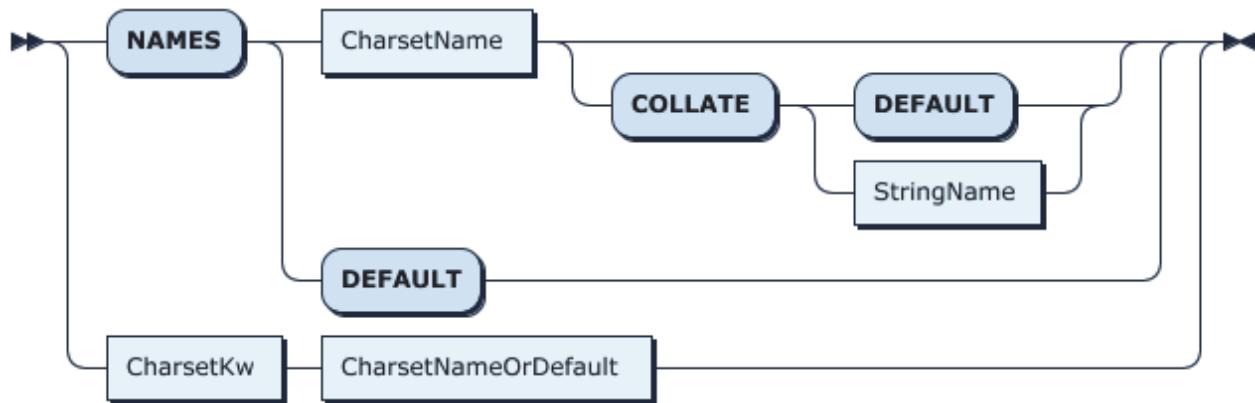


Figure 232: VariableAssignment

11.5.2.77.2 Examples

Get the value of `sql_mode`.

```
mysql> SHOW GLOBAL VARIABLES LIKE 'sql_mode';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| sql_mode      | ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,
|                  NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_AUTO_CREATE_USER,
|                  NO_ENGINE_SUBSTITUTION |
+-----+-----+
1 row in set (0.00 sec)

mysql> SHOW SESSION VARIABLES LIKE 'sql_mode';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| sql_mode      | ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,
|                  NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_AUTO_CREATE_USER,
|                  NO_ENGINE_SUBSTITUTION |
+-----+-----+
1 row in set (0.00 sec)
```

Update the value of `sql_mode` globally. If you check the value of `SQL_mode` after the update, you can see that the value of `SESSION` level has not been updated:

```
mysql> SET GLOBAL sql_mode = 'STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER';
Query OK, 0 rows affected (0.03 sec)

mysql> SHOW GLOBAL VARIABLES LIKE 'sql_mode';
```

```
+-----+
| Variable_name | Value
+-----+
| sql_mode      | STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER |
+-----+
1 row in set (0.00 sec)

mysql> SHOW SESSION VARIABLES LIKE 'sql_mode';
+-----+
| Variable_name | Value
|-----|
| sql_mode      | ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,
|                   NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_AUTO_CREATE_USER,
|                   NO_ENGINE_SUBSTITUTION |
+-----+
|-----|
|-----|
1 row in set (0.00 sec)
```

Using `SET SESSION` takes effect immediately:

```
mysql> SET SESSION sql_mode = 'STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER';
Query OK, 0 rows affected (0.01 sec)

mysql> SHOW SESSION VARIABLES LIKE 'sql_mode';
+-----+
| Variable_name | Value
+-----+
| sql_mode      | STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER |
+-----+
1 row in set (0.00 sec)
```

11.5.2.77.3 MySQL compatibility

The following behavior differences apply:

- Changes made with `SET GLOBAL` will be propagated to all TiDB instances in the cluster. This differs from MySQL, where changes do not propagate to replicas.

- TiDB presents several variables as both readable and settable. This is required for MySQL compatibility, because it is common for both applications and connectors to read MySQL variables. For example: JDBC connectors both read and set query cache settings, despite not relying on the behavior.
- Changes made with `SET GLOBAL` will persist through TiDB server restarts. This means that `SET GLOBAL` in TiDB behaves more similar to `SET PERSIST` as available in MySQL 8.0 and above.

11.5.2.77.4 See also

- [SHOW \[GLOBAL|SESSION\] VARIABLES](#)

11.5.2.78 SHOW ANALYZE STATUS

The `SHOW ANALYZE STATUS` statement shows the statistics collection tasks being executed by TiDB and a limited number of historical task records.

11.5.2.78.1 Synopsis

`ShowStmt:`

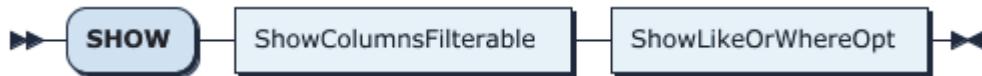


Figure 233: `ShowStmt`

`ShowTargetFilterable:`

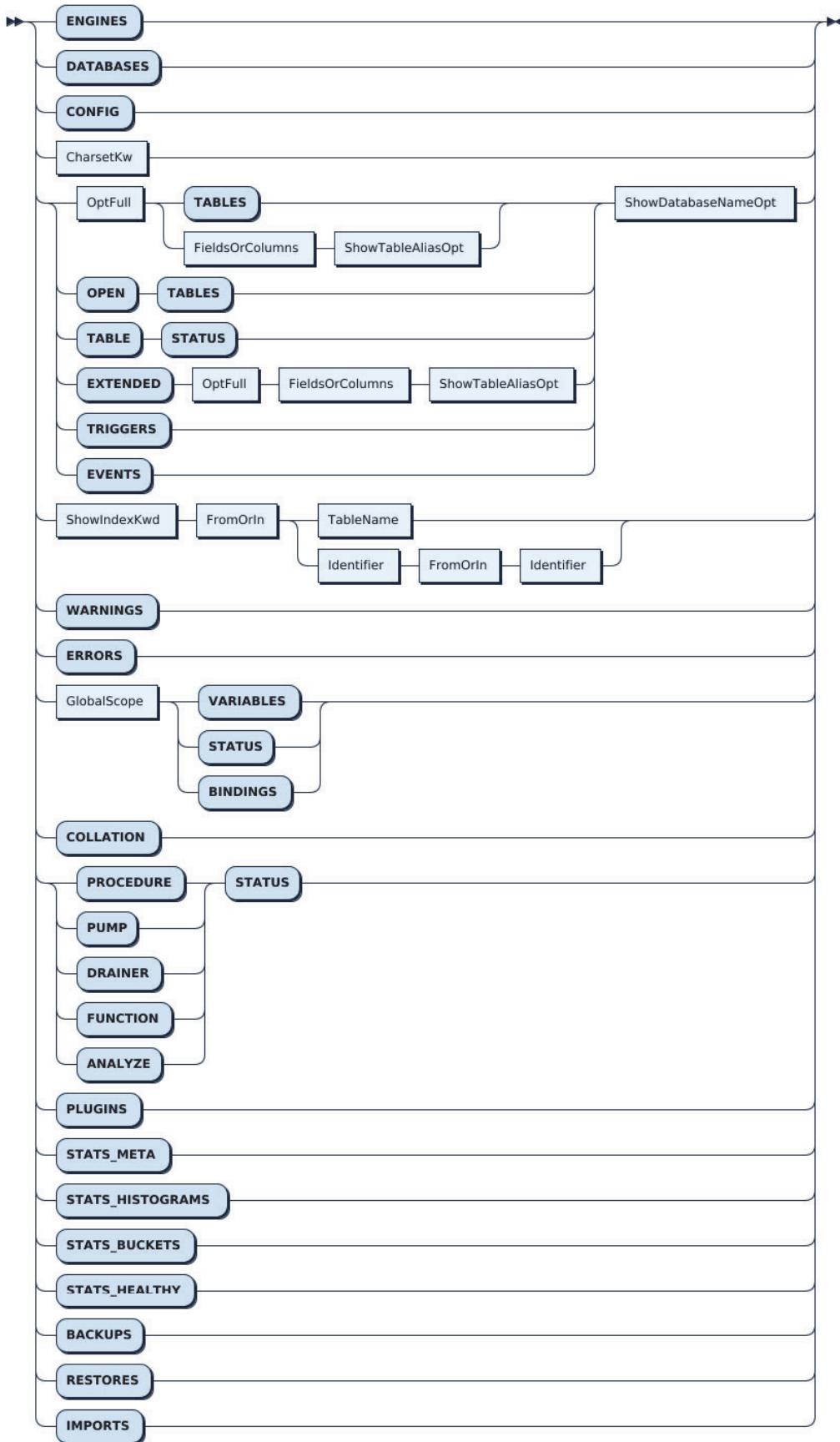


Figure 234: ShowTargetFilterable

11.5.2.78.2 Examples

```
create table t(x int, index idx(x)) partition by hash(x) partition 4;
analyze table t;
show analyze status;
```

Table_schema	Table_name	Partition_name	Job_info	Processed_rows
Start_time		State		
test	t	p1	analyze columns	0
		↪ 2020-05-25 17:23:55	finished	
test	t	p0	analyze columns	0
		↪ 2020-05-25 17:23:55	finished	
test	t	p0	analyze index idx	0
		↪ 2020-05-25 17:23:55	finished	
test	t	p1	analyze index idx	0
		↪ 2020-05-25 17:23:55	finished	
test	t	p2	analyze index idx	0
		↪ 2020-05-25 17:23:55	finished	
test	t	p3	analyze index idx	0
		↪ 2020-05-25 17:23:55	finished	
test	t	p3	analyze columns	0
		↪ 2020-05-25 17:23:55	finished	
test	t	p2	analyze columns	0
		↪ 2020-05-25 17:23:55	finished	

8 rows in set (0.00 sec)

11.5.2.78.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.78.4 See also

- [ANALYZE_STATUS table](#)

11.5.2.79 SHOW [BACKUPS|RESTORES]

These statements show a list of all queued, running and recently finished **BACKUP** and **RESTORE** tasks that were executed on a TiDB instance.

Both statements require SUPER privilege to run.

Use **SHOW BACKUPS** to query BACKUP tasks and use **SHOW RESTORES** to query RESTORE tasks.

Backups and restores that were started with the **br** commandline tool are not shown.

11.5.2.79.1 Synopsis

```
ShowBRIEStmt ::=  
    "SHOW" ("BACKUPS" | "RESTORES") ShowLikeOrWhere?  
  
ShowLikeOrWhere ::=  
    "LIKE" SimpleExpr  
  | "WHERE" Expression
```

11.5.2.79.2 Examples

In one connection, execute the following statement:

```
BACKUP DATABASE `test` TO 's3://example-bucket/backup-01/?region=us-west-1';
```

Before the backup completes, run **SHOW BACKUPS** in a new connection:

```
SHOW BACKUPS;
```

Destination	State	Progress	Queue_time	Execution_time	Finish_time	Connection	Message
s3://example-bucket/backup-01/	Backup	98.38	2020-04-12 23:09:03	2020-04-12 23:09:25	NULL	4	NULL

1 row in set (0.00 sec)

The first row of the result above is described as follows:

Column	Description
Destination	The ↪ destination URL (with all parameters stripped to avoid leaking secret keys)
State	State of the task
Progress	Estimated progress in the current state as a percentage
Queue_time	When the ↪ task was queued
Execution_time	When the ↪ task was started; the value is 0000-00-00 ↪ ↪ 00:00:00 ↪ for queueing tasks
Finish_time	The ↪ timestamp when the task finished; the value is 0000-00-00 ↪ ↪ 00:00:00 ↪ for queueing and running tasks

Column	Description
Connection	Connection ID running this task
Message	Message with details

The possible states are:

State	Description
Backup	Making a backup
Wait	Waiting for execution
Checksum	Running a checksum operation

The connection ID can be used to cancel a backup/restore task via the `KILL TIDB QUERY` statement.

```
KILL TIDB QUERY 4;
```

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

Filtering

Use the `LIKE` clause to filter out tasks by matching the destination URL against a wildcard expression.

```
SHOW BACKUPS LIKE 's3://%';
```

Use the `WHERE` clause to filter by columns.

```
SHOW BACKUPS WHERE `Progress` < 25.0;
```

11.5.2.79.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.79.4 See also

- [BACKUP](#)
- [RESTORE](#)

11.5.2.80 SHOW [GLOBAL|SESSION] BINDINGS

The `SHOW BINDINGS` statement is used to display information about created SQL bindings. A `BINDING` can be on either a `GLOBAL` or `SESSION` basis. The default is `SESSION`.

11.5.2.80.1 Synopsis

ShowStmt:

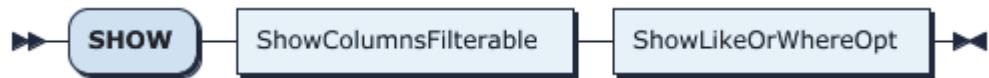


Figure 235: ShowStmt

ShowTargetFilterable:

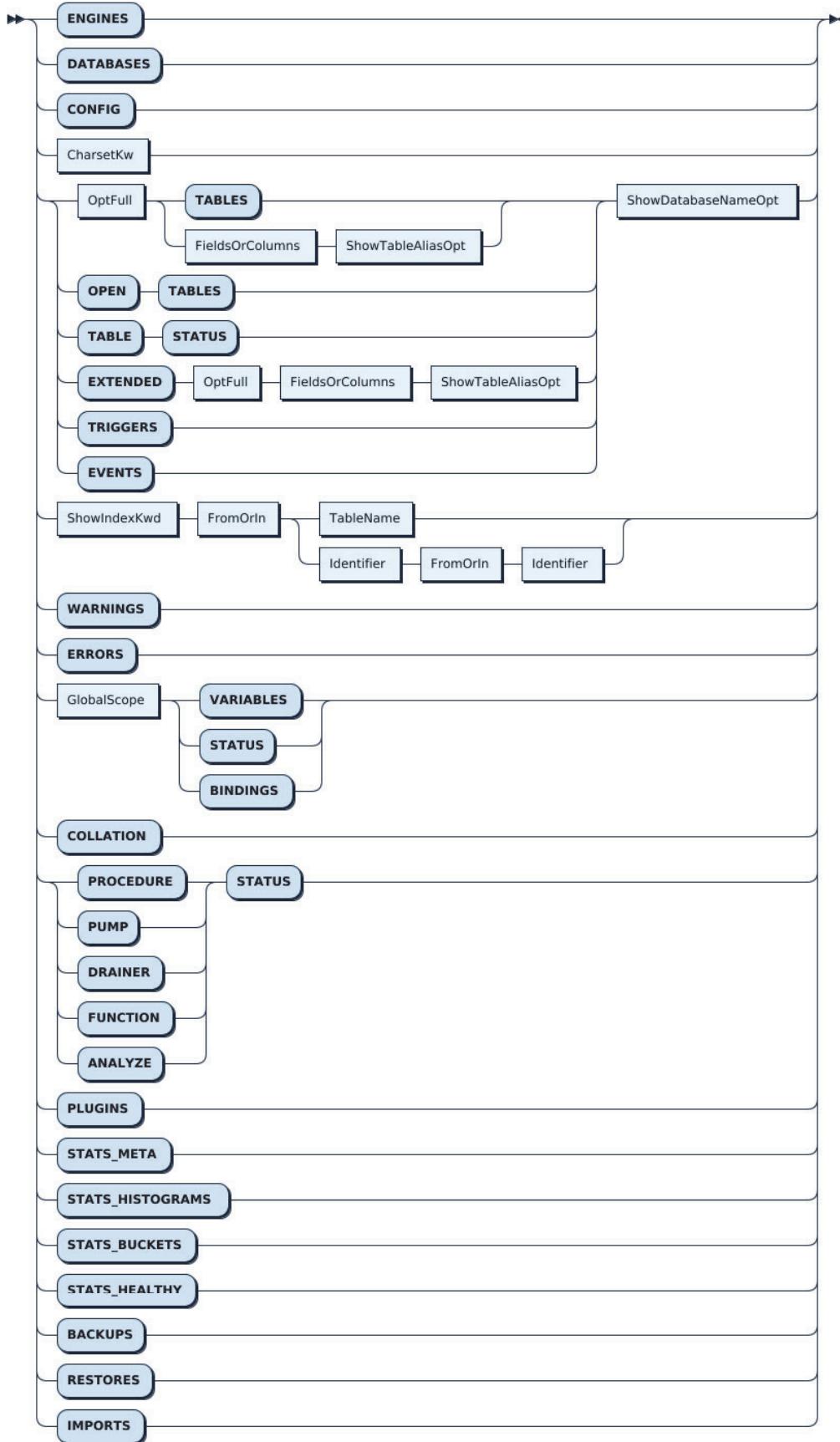


Figure 236: ShowTargetFilterable
1676

GlobalScope:

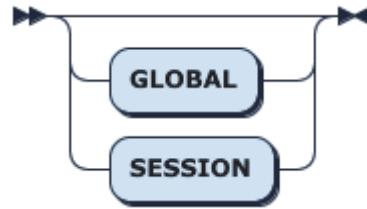


Figure 237: GlobalScope

ShowLikeOrWhereOpt

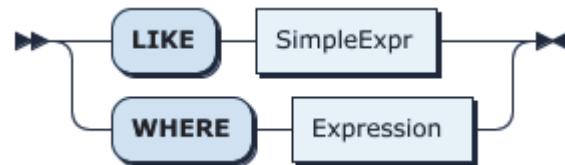


Figure 238: ShowLikeOrWhereOpt

11.5.2.80.2 Syntax description

```
SHOW [GLOBAL | SESSION] BINDINGS [ShowLikeOrWhereOpt];
```

This statement outputs the execution plan bindings at the GLOBAL or SESSION level. The default scope is SESSION. Currently SHOW BINDINGS outputs eight columns, as shown below:

Column	
Name	Description
original_sql	Original SQL statement after parameterization
bind_sql	Bound SQL statement with hints
default_db	Default database
status	Status including 'Using', 'Deleted', 'Invalid', 'Rejected', and 'Pending verification'

Column Name	Description
create_time	Created time
update_time	Updated time
charset	Character set
collation	Sorting rule
source	The way in which a binding is created, including <code>manual</code> (created by the <code>create</code> → [global] → <code>binding</code> SQL statement), <code>capture</code> (captured automatically by TiDB), and <code>evolve</code> (evolved automatically by TiDB)

11.5.2.80.3 Examples

```
mysql> CREATE TABLE t1 (
    -> id INT NOT NULL PRIMARY KEY auto_increment,
    -> b INT NOT NULL,
    -> pad VARBINARY(255),
    -> INDEX(b)
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    -> FROM dual;
Query OK, 1 row affected (0.01 sec)
Records: 1 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
    -> FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 1 row affected (0.00 sec)
Records: 1 Duplicates: 0 Warnings: 0
```

```

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
   ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 8 rows affected (0.00 sec)
Records: 8 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
   ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 1000 rows affected (0.04 sec)
Records: 1000 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
   ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 100000 rows affected (1.74 sec)
Records: 100000 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
   ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 100000 rows affected (2.15 sec)
Records: 100000 Duplicates: 0 Warnings: 0

mysql> INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255)
   ↪ FROM t1 a JOIN t1 b JOIN t1 c LIMIT 100000;
Query OK, 100000 rows affected (2.64 sec)
Records: 100000 Duplicates: 0 Warnings: 0

mysql> SELECT SLEEP(1);
+-----+
| SLEEP(1) |
+-----+
|      0   |
+-----+
1 row in set (1.00 sec)

mysql> ANALYZE TABLE t1;
Query OK, 0 rows affected (1.33 sec)

mysql> EXPLAIN ANALYZE SELECT * FROM t1 WHERE b = 123;
+-----+
| id          | estRows | actRows | task | access object
| execution info                               | operator info
|           | memory    | disk    |
+-----+

```

```

→ -----
→
| IndexLookUp_10           | 583.00 | 297    | root      |
→                         | time:10.545072ms, loops:2, rpc num: 1, rpc time
→ :398.359µs, proc keys:297 |                   | 109.1484375 KB | N/
→ A |
|
| -IndexRangeScan_8(Build) | 583.00 | 297    | cop[tikv] | table:t1, index
→ :b(b) | time:0s, loops:4
→ range:[123,123], keep order:false | N/A      | N/A |
|
| -TableRowIDScan_9(Probe)  | 583.00 | 297    | cop[tikv] | table:t1
→           | time:12ms, loops:4
→                           | keep order:false
→           | N/A          | N/A |
+--+
→ -----
→
3 rows in set (0.02 sec)

```

```

mysql> CREATE SESSION BINDING FOR
-> SELECT * FROM t1 WHERE b = 123
-> USING
-> SELECT * FROM t1 IGNORE INDEX (b) WHERE b = 123;
Query OK, 0 rows affected (0.00 sec)

```

```

mysql> EXPLAIN ANALYZE SELECT * FROM t1 WHERE b = 123;
+-
→ -----
→
| id           | estRows | actRows | task      | access object |
→ execution info
→ operator info | memory   | disk    |
+--+
→ -----
→
| TableReader_7 | 583.00  | 297    | root      |               | time
→ :222.32506ms, loops:2, rpc num: 1, rpc time:222.078952ms, proc keys
→ :301010 | data:Selection_6 | 88.6640625 KB | N/A |
|
| -Selection_6  | 583.00  | 297    | cop[tikv] |               | time
→ :224ms, loops:298
→ test.t1.b, 123) | N/A      | N/A |
|
| -TableFullScan_5 | 301010.00 | 301010 | cop[tikv] | table:t1 | time
→ :220ms, loops:298
→ keep order:false | N/A      | N/A |
+--+
→ -----
→

```

```
↪
3 rows in set (0.22 sec)

mysql> SHOW SESSION BINDINGS\G
***** 1. row *****
Original_sql: select * from t1 where b = ?
Bind_sql: SELECT * FROM t1 IGNORE INDEX (b) WHERE b = 123
Default_db: test
Status: using
Create_time: 2020-05-22 14:38:03.456
Update_time: 2020-05-22 14:38:03.456
Charset: utf8mb4
Collation: utf8mb4_0900_ai_ci
1 row in set (0.00 sec)
```

11.5.2.80.4 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.80.5 See also

- CREATE [GLOBAL|SESSION] BINDING
- DROP [GLOBAL|SESSION] BINDING
- ANALYZE TABLE
- Optimizer Hints
- SQL Plan Management

11.5.2.81 SHOW BUILTINS

SHOW BUILTINS is used to list all supported builtin functions in TiDB.

11.5.2.81.1 Synopsis

ShowBuiltinsStmt:



Figure 239: ShowBuiltinsStmt

11.5.2.81.2 Examples

```
SHOW BUILTINS;
```

-----+ Supported_builtin_functions -----+
abs
acos
adddate
addtime
aes_decrypt
aes_encrypt
and
any_value
ascii
asin
atan
atan2
benchmark
bin
bit_count
bit_length
bitand
bitneg
bitor
bitxor
case
ceil
ceiling
char_func
char_length
character_length
charset
coalesce
coercibility
collation
compress
concat
concat_ws
connection_id
conv
convert
convert_tz
cos
cot
crc32
curdate

current_date	
current_role	
current_time	
current_timestamp	
current_user	
curtime	
database	
date	
date_add	
date_format	
date_sub	
datediff	
day	
dayname	
dayofmonth	
dayofweek	
dayofyear	
decode	
default_func	
degrees	
des_decrypt	
des_encrypt	
div	
elt	
encode	
encrypt	
eq	
exp	
export_set	
extract	
field	
find_in_set	
floor	
format	
format_bytes	
format_nano_time	
found_rows	
from_base64	
from_days	
from_unixtime	
ge	
get_format	
get_lock	
getparam	
getvar	

greatest	
gt	
hex	
hour	
if	
ifnull	
in	
inet6_aton	
inet6_ntoa	
inet_aton	
inet_ntoa	
insert_func	
instr	
intdiv	
interval	
is_free_lock	
is_ipv4	
is_ipv4_compat	
is_ipv4_mapped	
is_ipv6	
is_used_lock	
isfalse	
isnull	
istrue	
json_array	
json_array_append	
json_array_insert	
json_contains	
json_contains_path	
json_depth	
json_extract	
json_insert	
json_keys	
json_length	
json_merge	
json_merge_patch	
json_merge_preserve	
json_object	
json_pretty	
json_quote	
json_remove	
json_replace	
json_search	
json_set	
json_storage_size	

json_type	
json_unquote	
json_valid	
last_day	
last_insert_id	
lastval	
lcase	
le	
least	
left	
leftshift	
length	
like	
ln	
load_file	
localtime	
localtimestamp	
locate	
log	
log10	
log2	
lower	
lpad	
lt	
ltrim	
make_set	
makedate	
maketime	
master_pos_wait	
md5	
microsecond	
mid	
minus	
minute	
mod	
month	
monthname	
mul	
name_const	
ne	
nextval	
not	
now	
nulleq	
oct	

octet_length	
old_password	
or	
ord	
password_func	
period_add	
period_diff	
pi	
plus	
position	
pow	
power	
quarter	
quote	
radians	
rand	
random_bytes	
regexp	
release_all_locks	
release_lock	
repeat	
replace	
reverse	
right	
rightshift	
round	
row_count	
rpad	
rtrim	
schema	
sec_to_time	
second	
session_user	
setval	
setvar	
sha	
sha1	
sha2	
sign	
sin	
sleep	
space	
sqrt	
str_to_date	
strcmp	

subdate	
substr	
substring	
substring_index	
subtime	
sysdate	
system_user	
tan	
tidb_decode_key	
tidb_decode_plan	
tidb_is_ddl_owner	
tidb_parse_tso	
tidb_version	
time	
time_format	
time_to_sec	
timediff	
timestamp	
timestampadd	
timestampdiff	
to_base64	
to_days	
to_seconds	
trim	
truncate	
ucase	
unaryminus	
uncompress	
uncompressed_length	
unhex	
unix_timestamp	
upper	
user	
utc_date	
utc_time	
utc_timestamp	
uuid	
uuid_short	
validate_password_strength	
version	
week	
weekday	
weekofyear	
weight_string	
xor	

```

| year           |
| yearweek      |
+-----+
268 rows in set (0.00 sec)

```

11.5.2.81.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.82 SHOW CHARACTER SET

This statement provides a static list of available character sets in TiDB. The output does not reflect any attributes of the current connection or user.

11.5.2.82.1 Synopsis

ShowCharsetStmt:



Figure 240: ShowCharsetStmt

CharsetKw:

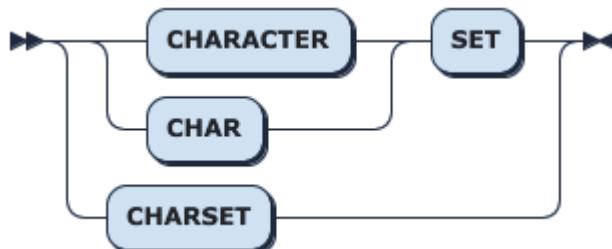


Figure 241: CharsetKw

11.5.2.82.2 Examples

```

mysql> SHOW CHARACTER SET;
+-----+-----+-----+-----+
| Charset | Description | Default collation | Maxlen |
+-----+-----+-----+-----+
| utf8    | UTF-8 Unicode | utf8_bin          |      3 |
| utf8mb4 | UTF-8 Unicode | utf8mb4_bin       |      4 |
| ascii   | US ASCII      | ascii_bin         |      1 |
| latin1  | Latin1        | latin1_bin        |      1 |

```

binary binary binary 1
+-----+-----+-----+-----+
5 rows in set (0.00 sec)

11.5.2.82.3 MySQL compatibility

The usage of this statement is understood to be fully compatible with MySQL. However, charsets in TiDB may have different default collations compared with MySQL. For details, refer to [Compatibility with MySQL](#). Any other compatibility differences should be [reported via an issue](#) on GitHub.

11.5.2.82.4 See also

- [SHOW COLLATION](#)
- [Character Set and Collation](#)

11.5.2.83 SHOW COLLATION

This statement provides a static list of collations, and is included to provide compatibility with MySQL client libraries.

Note:

Results of `SHOW COLLATION` vary when the “new collation framework” is enabled. For new collation framework details, refer to [Character Set and Collation](#).

11.5.2.83.1 Synopsis

`ShowCollationStmt:`

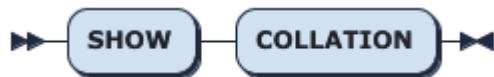


Figure 242: ShowCollationStmt

11.5.2.83.2 Examples

When new collation framework is disabled, only binary collations are displayed.

```
mysql> SHOW COLLATION;
+-----+-----+-----+-----+-----+
| Collation | Charset | Id | Default | Compiled | Sortlen |
+-----+-----+-----+-----+-----+
| utf8mb4_bin | utf8mb4 | 46 | Yes | Yes | 1 |
| latin1_bin | latin1 | 47 | Yes | Yes | 1 |
| binary | binary | 63 | Yes | Yes | 1 |
| ascii_bin | ascii | 65 | Yes | Yes | 1 |
| utf8_bin | utf8 | 83 | Yes | Yes | 1 |
+-----+-----+-----+-----+-----+
5 rows in set (0.02 sec)
```

When new collation framework is enabled, `utf8_general_ci` and `utf8mb4_general_ci` are additionally supported.

```
mysql> SHOW COLLATION;
+-----+-----+-----+-----+-----+
| Collation | Charset | Id | Default | Compiled | Sortlen |
+-----+-----+-----+-----+-----+
| utf8mb4_bin | utf8mb4 | 46 | Yes | Yes | 1 |
| latin1_bin | latin1 | 47 | Yes | Yes | 1 |
| binary | binary | 63 | Yes | Yes | 1 |
| ascii_bin | ascii | 65 | Yes | Yes | 1 |
| utf8_bin | utf8 | 83 | Yes | Yes | 1 |
| utf8_general_ci | utf8 | 33 | Yes | Yes | 1 |
| utf8mb4_general_ci | utf8 | 45 | Yes | Yes | 1 |
+-----+-----+-----+-----+-----+
7 rows in set (0.02 sec)
```

11.5.2.83.3 MySQL compatibility

The usage of this statement is understood to be fully compatible with MySQL. However, charsets in TiDB may have different default collations compared with MySQL. For details, refer to [Compatibility with MySQL](#). Any other compatibility differences should be [reported via an issue on GitHub](#).

11.5.2.83.4 See also

- [SHOW CHARACTER SET](#)
- [Character Set and Collation](#)

11.5.2.84 SHOW [FULL] COLUMNS FROM

The statement `SHOW [FULL] COLUMNS FROM <table_name>` describes the columns of a table or view in a useful tabular format. The optional keyword `FULL` displays the privileges the current user has to that column, and the `comment` from the table definition.

The statements `SHOW [FULL] FIELDS FROM <table_name>`, `DESC <table_name>`, `DESCRIBE <table_name>`, and `EXPLAIN <table_name>` are aliases of this statement.

Note:

`DESC TABLE <table_name>`, `DESCRIBE TABLE <table_name>`, and `EXPLAIN TABLE <table_name>` are not equivalent to the above statements. They are aliases of `DESC SELECT * FROM <table_name>`.

11.5.2.84.1 Synopsis

ShowStmt:



Figure 243: ShowStmt

ShowColumnsFilterable:



Figure 244: ShowColumnsFilterable

OptFull:



Figure 245: OptFull

FieldsOrColumns:

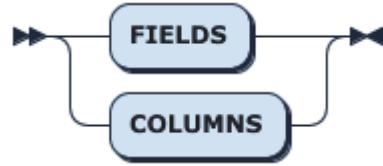


Figure 246: FieldsOrColumns

ShowTableAliasOpt:



Figure 247: ShowTableAliasOpt

FromOrIn:

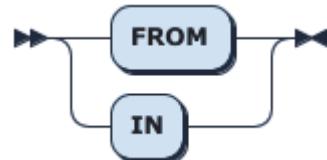


Figure 248: FromOrIn

TableName:

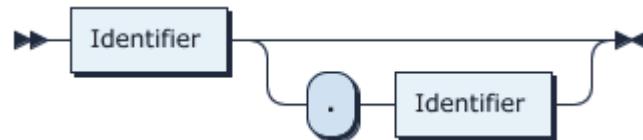


Figure 249: TableName

ShowDatabaseNameOpt:



Figure 250: ShowDatabaseNameOpt

DBName:



Figure 251: DBName

ShowLikeOrWhereOpt:

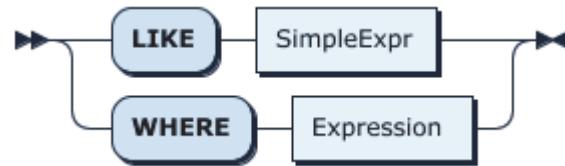


Figure 252: ShowLikeOrWhereOpt

11.5.2.84.2 Examples

```

mysql> create view v1 as select 1;
Query OK, 0 rows affected (0.11 sec)

mysql> show columns from v1;
+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+
| 1     | bigint(1) | YES |     | NULL    |       |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> desc v1;
+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+
| 1     | bigint(1) | YES |     | NULL    |       |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> describe v1;
+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+
| 1     | bigint(1) | YES |     | NULL    |       |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> explain v1;
+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+
| 1     | bigint(1) | YES |     | NULL    |       |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
  
```

```

mysql> show fields from v1;
+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+
| 1     | bigint(1) | YES  |      | NULL    |       |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> show full columns from v1;
+-----+
| Field | Type   | Collation | Null | Key | Default | Extra | Privileges
|       |          |           |      |     |         |       | Comment |
+-----+
| 1     | bigint(1) | NULL    | YES  |      | NULL    |       | select,insert,
|       |          |          |      |     |         |       | update,references |
+-----+
|       |          |           |      |     |         |       |
|       |          |           |      |     |         |       |
1 row in set (0.00 sec)

mysql> show full columns from mysql.user;
+-----+
| Field           | Type   | Collation | Null | Key | Default |
|                 |          |           |      |     |         |
| Extra          | Privileges |           |      |     |         |
|                 |          |           |      |     |         |
+-----+
| Host           | char(64) | utf8mb4_bin | NO  | PRI | NULL   |
|                 |          |             |      |     |         |
| select,insert,update,references |      |
| User           | char(32)  | utf8mb4_bin | NO  | PRI | NULL   |
|                 |          |             |      |     |         |
| select,insert,update,references |      |
| authentication_string | text  | utf8mb4_bin | YES |      | NULL   |
|                 |          |             |      |     |         |
| select,insert,update,references |      |
| Select_priv     | enum('N','Y') | utf8mb4_bin | NO  |      | N      |
|                 |          |             |      |     |         |
| select,insert,update,references |      |
| Insert_priv     | enum('N','Y') | utf8mb4_bin | NO  |      | N      |
|                 |          |             |      |     |         |
| select,insert,update,references |      |
| Update_priv     | enum('N','Y') | utf8mb4_bin | NO  |      | N      |
|                 |          |             |      |     |         |

```

	→ select,insert,update,references						
Delete_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Create_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Drop_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Process_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Grant_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
References_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Alter_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Show_db_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Super_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Create_tmp_table_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Lock_tables_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Execute_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Create_view_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Show_view_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Create_routine_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Alter_routine_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Index_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Create_user_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Event_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Trigger_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Create_role_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							
Drop_role_priv	enum('N','Y') utf8mb4_bin NO N						
→ select,insert,update,references							

Account_locked enum('N','Y') utf8mb4_bin NO N
↳ <code>select,insert,update,references</code>
Shutdown_priv enum('N','Y') utf8mb4_bin NO N
↳ <code>select,insert,update,references</code>
Reload_priv enum('N','Y') utf8mb4_bin NO N
↳ <code>select,insert,update,references</code>
FILE_priv enum('N','Y') utf8mb4_bin NO N
↳ <code>select,insert,update,references</code>
Config_priv enum('N','Y') utf8mb4_bin NO N
↳ <code>select,insert,update,references</code>
+--
↳ -----+-----+-----+-----+-----+
↳
33 rows in set (0.01 sec)

11.5.2.84.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.84.4 See also

- [SHOW CREATE TABLE](#)

11.5.2.85 SHOW CONFIG

Warning:

This feature is currently an experimental feature. It is not recommended to use this feature in the production environment.

The `SHOW CONFIG` statement is used to show the current configuration of various components of TiDB. Note that the configuration and system variables act on different dimensions and should not be mixed up. If you want to obtain the system variable information, use the [SHOW VARIABLES](#) syntax.

11.5.2.85.1 Synopsis

ShowStmt:

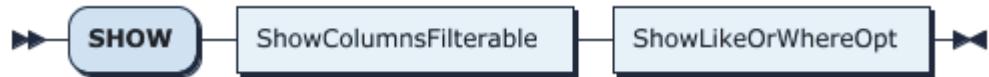


Figure 253: ShowStmt

ShowTargetFilterable:

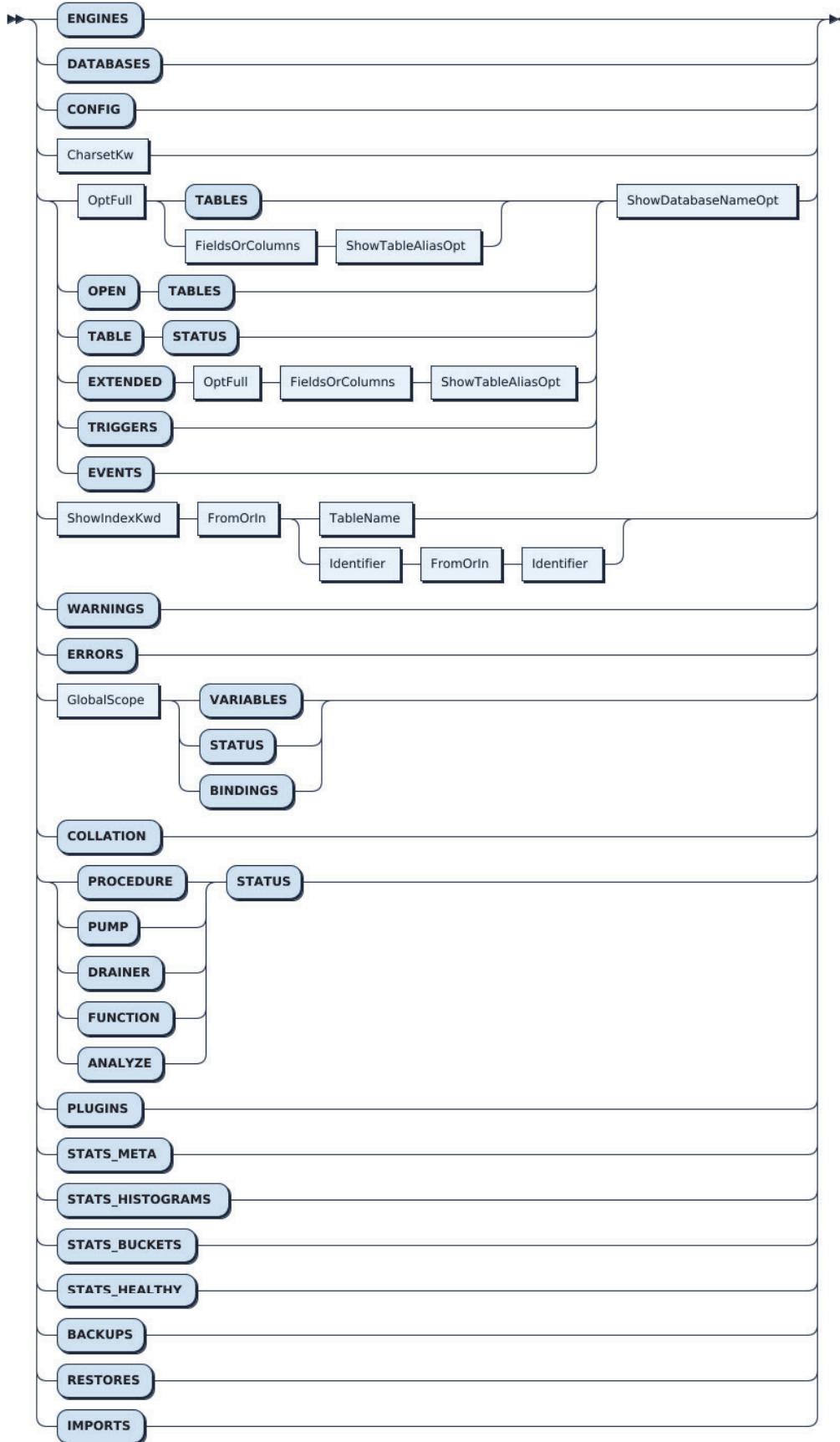


Figure 254: ShowTargetFilterable
1698

11.5.2.85.2 Examples

Show all configurations:

```
SHOW CONFIG;
```

Type	Instance	Name	Value
tidb	127.0.0.1:4000	advertise-address	127.0.0.1
tidb	127.0.0.1:4000	binlog.binlog-socket	
tidb	127.0.0.1:4000	binlog.enable	false
...			

120 rows in set (0.01 sec)

Show the configuration where the type is tidb:

```
SHOW CONFIG WHERE type = 'tidb' AND name = 'advertise-address';
```

Type	Instance	Name	Value
tidb	127.0.0.1:4000	advertise-address	127.0.0.1

1 row in set (0.05 sec)

You can also use the LIKE clause to show the configuration where the type is tidb:

```
SHOW CONFIG LIKE 'tidb';
```

Type	Instance	Name	Value
tidb	127.0.0.1:4000	advertise-address	127.0.0.1
tidb	127.0.0.1:4000	binlog.binlog-socket	

```
| tidb | 127.0.0.1:4000 | binlog.enable | false
  ↵
...
40 rows in set (0.01 sec)
```

11.5.2.85.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.85.4 See also

- [SHOW VARIABLES](#)

11.5.2.86 SHOW CREATE PLACEMENT POLICY

Warning:

Placement Rules in SQL is an experimental feature. The syntax might change before its GA, and there might also be bugs.

If you understand the risks, you can enable this experiment feature by executing `SET GLOBAL tidb_enable_alter_placement = 1;`.

`SHOW CREATE PLACEMENT POLICY` is used to show the definition of a placement policy. This can be used to see the current definition of a placement policy and recreate it in another TiDB cluster.

11.5.2.86.1 Synopsis

```
"SHOW" "CREATE" "PLACEMENT" "POLICY" PolicyName
PolicyName ::= Identifier
```

11.5.2.86.2 Examples

```
CREATE PLACEMENT POLICY p1 PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us
  ↵ -west-1" FOLLOWERS=4;
CREATE TABLE t1 (a INT) PLACEMENT POLICY=p1;
SHOW CREATE PLACEMENT POLICY p1\G
```

```

Query OK, 0 rows affected (0.08 sec)

Query OK, 0 rows affected (0.10 sec)

*****
 1. row *****
Policy: p1
Create Policy: CREATE PLACEMENT POLICY `p1` PRIMARY_REGION="us-east-1"
  ↪ REGIONS="us-east-1,us-west-1" FOLLOWERS=4
1 row in set (0.00 sec)

```

11.5.2.86.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.86.4 See also

- [Placement Rules in SQL](#)
- [SHOW PLACEMENT](#)
- [CREATE PLACEMENT POLICY](#)
- [ALTER PLACEMENT POLICY](#)
- [DROP PLACEMENT POLICY](#)

11.5.2.87 SHOW CREATE SEQUENCE

The `SHOW CREATE SEQUENCE` shows the detailed information of a sequence, which is similar to `SHOW CREATE TABLE`.

11.5.2.87.1 Synopsis

`ShowCreateSequenceStmt:`



Figure 255: ShowCreateSequenceStmt

`TableName:`

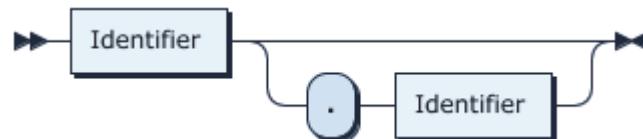


Figure 256: TableName

11.5.2.87.2 Examples

```
CREATE SEQUENCE seq;
```

```
Query OK, 0 rows affected (0.03 sec)
```

```
SHOW CREATE SEQUENCE seq;
```

```
+-----+
| Table | Create Table
+-----+
| seq   | CREATE SEQUENCE `seq` start with 1 minvalue 1 maxvalue
      |   ↵ 9223372036854775806 increment by 1 cache 1000 nocycle ENGINE=InnoDB |
+-----+
|       |
1 row in set (0.00 sec)
```

11.5.2.87.3 MySQL compatibility

This statement is a TiDB extension. The implementation is modeled on sequences available in MariaDB.

11.5.2.87.4 See also

- [CREATE SEQUENCE](#)
- [DROP SEQUENCE](#)

11.5.2.88 SHOW CREATE TABLE

This statement shows the exact statement to recreate an existing table using SQL.

11.5.2.88.1 Synopsis

`ShowCreateTableStmt:`



Figure 257: ShowCreateTableStmt

`TableName:`

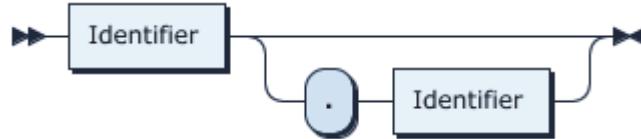


Figure 258: TableName

11.5.2.88.2 Examples

```

mysql> CREATE TABLE t1 (a INT);
Query OK, 0 rows affected (0.12 sec)

mysql> SHOW CREATE TABLE t1;
+--+
| Table | Create Table
+--+
| t1   | CREATE TABLE `t1` (
  `a` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin |
+--+
1 row in set (0.00 sec)

```

11.5.2.88.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.88.4 See also

- [CREATE TABLE](#)
- [DROP TABLE](#)
- [SHOW TABLES](#)
- [SHOW COLUMNS FROM](#)

11.5.2.89 SHOW CREATE USER

This statement shows how to re-create a user using the CREATE USER syntax.

11.5.2.89.1 Synopsis

ShowCreateUserStmt:



Figure 259: ShowCreateUserStmt

Username:

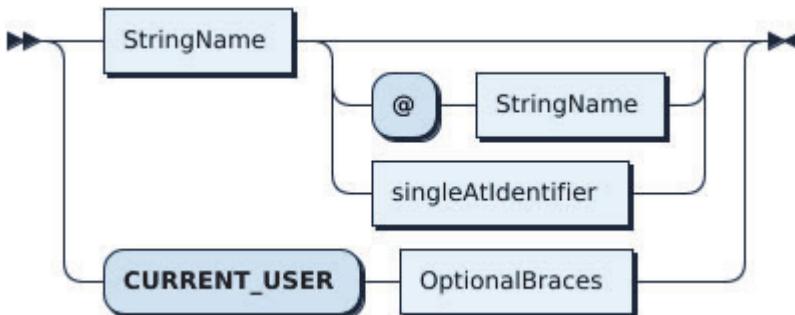


Figure 260: Username

11.5.2.89.2 Examples

```

mysql> SHOW CREATE USER 'root';
+--+
| CREATE USER for root@%
|   |
|   | CREATE USER 'root'@'%' IDENTIFIED WITH 'mysql_native_password' AS ''
|   |   REQUIRE NONE PASSWORD EXPIRE DEFAULT ACCOUNT UNLOCK |
+--+
|   |
|   |
1 row in set (0.00 sec)

mysql> SHOW GRANTS FOR 'root';
+-----+-----+
| Grants for root@%          |
+-----+-----+
| GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' |
+-----+-----+

```

```
+-----+
1 row in set (0.00 sec)
```

11.5.2.89.3 MySQL compatibility

- The output of SHOW CREATE USER is designed to match MySQL, but several of the CREATE options are not yet supported by TiDB. Not yet supported options will be parsed but ignored. See [security compatibility] for more details.

11.5.2.89.4 See also

- [CREATE USER](#)
- [SHOW GRANTS](#)
- [DROP USER](#)

11.5.2.90 SHOW DATABASES

This statement shows a list of databases that the current user has privileges to. Databases which the current user does not have access to will appear hidden from the list. The `information_schema` database always appears first in the list of databases.

`SHOW SCHEMAS` is an alias of this statement.

11.5.2.90.1 Synopsis

`ShowDatabasesStmt:`



Figure 261: ShowDatabasesStmt

`ShowLikeOrWhereOpt:`

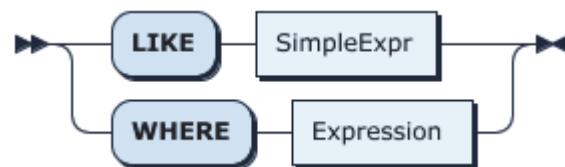


Figure 262: ShowLikeOrWhereOpt

11.5.2.90.2 Examples

```

mysql> SHOW DATABASES;
+-----+
| Database      |
+-----+
| INFORMATION_SCHEMA |
| PERFORMANCE_SCHEMA |
| mysql          |
| test           |
+-----+
4 rows in set (0.00 sec)

mysql> CREATE DATABASE mynewdb;
Query OK, 0 rows affected (0.10 sec)

mysql> SHOW DATABASES;
+-----+
| Database      |
+-----+
| INFORMATION_SCHEMA |
| PERFORMANCE_SCHEMA |
| mynewdb         |
| mysql          |
| test           |
+-----+
5 rows in set (0.00 sec)

```

11.5.2.90.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.90.4 See also

- [SHOW SCHEMAS](#)
- [DROP DATABASE](#)
- [CREATE DATABASE](#)

11.5.2.91 SHOW DRAINER STATUS

The `SHOW DRAINER STATUS` statement displays the status information for all Drainer nodes in the cluster.

11.5.2.91.1 Examples

```
SHOW DRAINER STATUS;
```

```
+--+
| NodeID | Address | State | Max_Commit_Ts | Update_Time |
+--+
| drainer1 | 127.0.0.3:8249 | Online | 408553768673342532 | 2019-05-01
|           | 00:00:03 |
+--+
| drainer2 | 127.0.0.4:8249 | Online | 408553768673345531 | 2019-05-01
|           | 00:00:04 |
+--+
2 rows in set (0.00 sec)
```

11.5.2.91.2 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.91.3 See also

- [SHOW PUMP STATUS](#)
- [CHANGE PUMP STATUS](#)
- [CHANGE DRAINER STATUS](#)

11.5.2.92 SHOW ENGINES

This statement is used to list all supported storage engines. The syntax is included only for compatibility with MySQL.

11.5.2.92.1 Synopsis

ShowEnginesStmt:

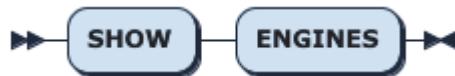


Figure 263: ShowEnginesStmt

```
SHOW ENGINES
```

11.5.2.92.2 Examples

```
mysql> SHOW ENGINES;
+----+-----+-----+
| Engine | Support | Comment          |
|        | Transactions | XA | Savepoints |
+----+-----+-----+
| InnoDB | DEFAULT | Supports transactions, row-level locking, and foreign keys | YES | YES | YES |
+----+-----+-----+
1 row in set (0.00 sec)
```

11.5.2.92.3 MySQL compatibility

- This statement will always only return InnoDB as the supported engine. Internally, TiDB will typically use TiKV as the storage engine.

11.5.2.93 SHOW ERRORS

This statement shows error(s) from previously executed statements. The error buffer is cleared as soon as a statement executes successfully. In which case, `SHOW ERRORS` will return an empty set.

The behavior of which statements generate errors vs. warnings is highly influenced by the current `sql_mode`.

11.5.2.93.1 Synopsis

`ShowErrorsStmt:`



Figure 264: ShowErrorsStmt

11.5.2.93.2 Examples

```

mysql> select invalid;
ERROR 1054 (42S22): Unknown column 'invalid' in 'field list'
mysql> create invalid;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual
    ↪ that corresponds to your TiDB version for the right syntax to use
    ↪ line 1 column 14 near "invalid"
mysql> SHOW ERRORS;
+---+
| Level | Code | Message
|       |      |
+---+
| Error | 1054 | Unknown column 'invalid' in 'field list'
|       |      |
| Error | 1064 | You have an error in your SQL syntax; check the manual
    ↪ that corresponds to your TiDB version for the right syntax to use
    ↪ line 1 column 14 near "invalid" |
+---+
|       |      |
2 rows in set (0.00 sec)

mysql> CREATE invalid2;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual
    ↪ that corresponds to your TiDB version for the right syntax to use
    ↪ line 1 column 15 near "invalid2"
mysql> SELECT 1;
+---+
| 1   |
+---+
|   1 |
+---+
1 row in set (0.00 sec)

mysql> SHOW ERRORS;
Empty set (0.00 sec)

```

11.5.2.93.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be [reported via an issue on GitHub](#).

11.5.2.93.4 See also

- [SHOW WARNINGS](#)

11.5.2.94 SHOW [FULL] FIELDS FROM

This statement is an alias to `SHOW [FULL] COLUMNS FROM`. It is included for compatibility with MySQL.

11.5.2.95 SHOW GRANTS

This statement shows a list of privileges associated with a user. As in MySQL, the `USAGE` privilege denotes the ability to login to TiDB.

11.5.2.95.1 Synopsis

`ShowGrantsStmt:`

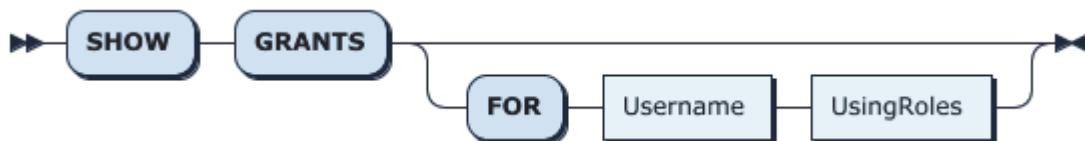


Figure 265: `ShowGrantsStmt`

`Username:`

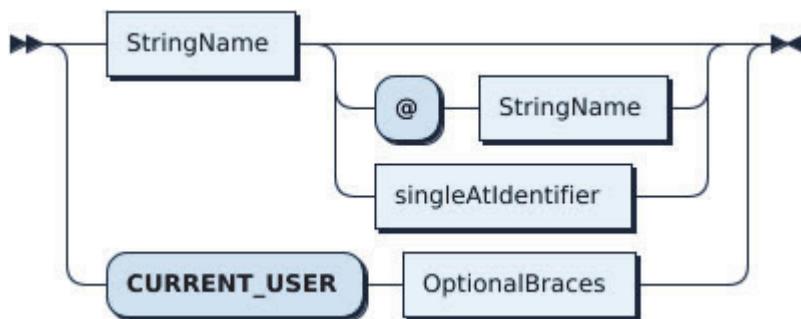


Figure 266: `Username`

`UsingRoles:`



Figure 267: UsingRoles

RolenameList:

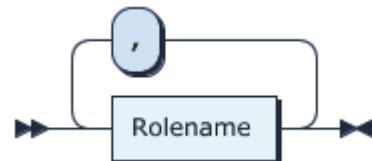


Figure 268: RolenameList

Rolename:

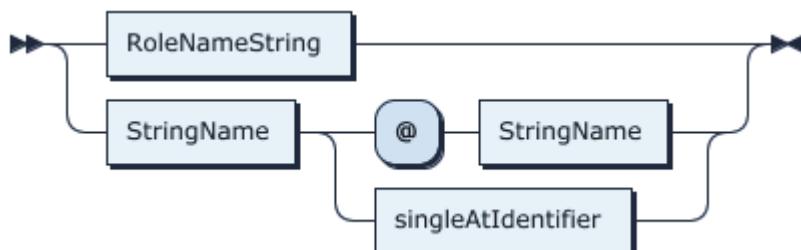


Figure 269: Rolename

11.5.2.95.2 Examples

```
mysql> SHOW GRANTS;
+-----+
| Grants for User |
+-----+
| GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' |
+-----+
1 row in set (0.00 sec)

mysql> SHOW GRANTS FOR 'u1';
ERROR 1141 (42000): There is no such grant defined for user 'u1' on host '%'
   ↗
mysql> CREATE USER u1;
Query OK, 1 row affected (0.04 sec)

mysql> GRANT SELECT ON test.* TO u1;
Query OK, 0 rows affected (0.04 sec)
```

```
mysql> SHOW GRANTS FOR u1;
+-----+
| Grants for u1@% |
+-----+
| GRANT USAGE ON *.* TO 'u1'@'%' |
| GRANT Select ON test.* TO 'u1'@'%' |
+-----+
2 rows in set (0.00 sec)
```

11.5.2.95.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.95.4 See also

- [SHOW CREATE USER](#)
- [GRANT](#)

11.5.2.96 SHOW INDEX [FROM|IN]

This statement is an alias to [SHOW INDEXES \[FROM|IN\]](#). It is included for compatibility with MySQL.

11.5.2.97 SHOW INDEXES [FROM|IN]

The statement `SHOW INDEXES [FROM|IN]` lists the indexes on a specified table. The statements `SHOW INDEX [FROM|IN]`, `SHOW KEYS [FROM|IN]` are aliases of this statement, and included for compatibility with MySQL.

11.5.2.97.1 Synopsis

`ShowIndexStmt:`

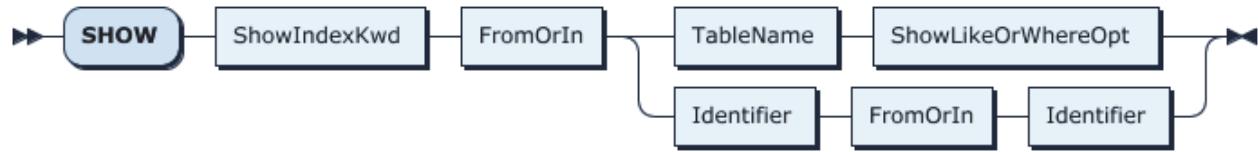


Figure 270: ShowIndexStmt

`ShowIndexKwd:`

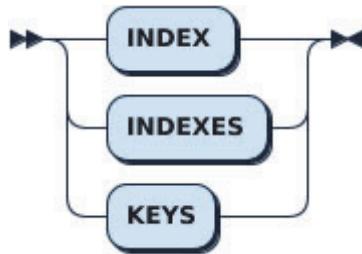


Figure 271: ShowIndexKwd

FromOrIn:

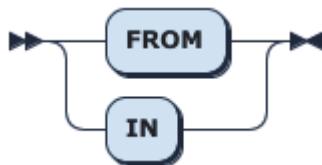


Figure 272: FromOrIn

TableName:

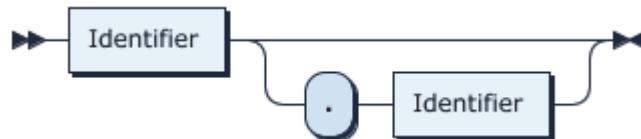


Figure 273: TableName

ShowLikeOrWhereOpt:

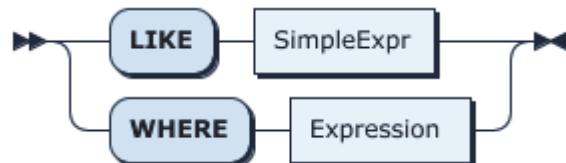


Figure 274: ShowLikeOrWhereOpt

11.5.2.97.2 Examples

```

mysql> CREATE TABLE t1 (id int not null primary key AUTO_INCREMENT, col1
   → INT, INDEX(col1));
Query OK, 0 rows affected (0.12 sec)

mysql> SHOW INDEXES FROM t1;

```

```
+--+
→ -----+-----+-----+-----+-----+
→
| Table | Non_unique | Key_name | Seq_in_index | Column_name | Collation |
→ Cardinality | Sub_part | Packed | Null | Index_type | Comment |
→ Index_comment | Visible | Expression |
+--+
→ -----+-----+-----+-----+-----+
→
| t1 | 0 | PRIMARY | 1 | id | A | 0
→ | NULL | NULL | | BTREE | | YES
| NULL |
| t1 | 1 | col1 | 1 | col1 | A | 0
→ | NULL | NULL | YES | BTREE | | YES
| NULL |
+--+
→ -----+-----+-----+-----+-----+
→
2 rows in set (0.00 sec)
```

mysql> SHOW INDEX FROM t1;

```
+--+
→ -----+-----+-----+-----+-----+
→
| Table | Non_unique | Key_name | Seq_in_index | Column_name | Collation |
→ Cardinality | Sub_part | Packed | Null | Index_type | Comment |
→ Index_comment | Visible | Expression |
+--+
→ -----+-----+-----+-----+-----+
→
| t1 | 0 | PRIMARY | 1 | id | A | 0
→ | NULL | NULL | | BTREE | | YES
| NULL |
| t1 | 1 | col1 | 1 | col1 | A | 0
→ | NULL | NULL | YES | BTREE | | YES
| NULL |
+--+
→ -----+-----+-----+-----+-----+
→
2 rows in set (0.00 sec)
```

mysql> SHOW KEYS FROM t1;

```
+--+
→ -----+-----+-----+-----+-----+
→
```

```
| Table | Non_unique | Key_name | Seq_in_index | Column_name | Collation | |
|       | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
|       | Index_comment | Visible | Expression |
+---+
|   → -----+-----+-----+-----+-----+
|   →
|   →
| t1 |      0 | PRIMARY |          1 | id          | A          | |
|   → |    NULL |    NULL |          | BTREE        |           |           |
|   |    NULL |          |          |             |           |           |
| t1 |      1 | col1    |          1 | col1        | A          |
|   → |    NULL |    NULL | YES | BTREE        |           |           |
|   |    NULL |          |          |             |           |           |
+---+
|   → -----+-----+-----+-----+-----+
|   →
|   →
2 rows in set (0.00 sec)
```

11.5.2.97.3 MySQL compatibility

The `Cardinality` column in MySQL shows the number of different values on the index. In TiDB, the `Cardinality` column always shows 0.

11.5.2.97.4 See also

- SHOW CREATE TABLE
 - DROP INDEX
 - CREATE INDEX

11.5.2.98 SHOW KEYS [FROM|IN]

This statement is an alias to `SHOW INDEXES [FROM|IN]`. It is included for compatibility with MySQL.

11.5.2.99 SHOW MASTER STATUS

The SHOW MASTER STATUS statement displays the latest TSO in the cluster.

11.5.2.99.1 Examples

SHOW MASTER STATUS;

1

```

+----+-----+-----+
| File      | Position          | Binlog_Do_DB | Binlog_Ignore_DB |
+----+-----+-----+
| tidb-binlog | 416916363252072450 |           |           |
+----+-----+-----+
1 row in set (0.00 sec)

```

11.5.2.99.2 MySQL compatibility

The output of `SHOW MASTER STATUS` is designed to match MySQL. However, the execution results are different in that the MySQL result is the binlog location information and the TiDB result is the latest TSO information.

11.5.2.99.3 See also

- [SHOW PUMP STATUS](#)
- [SHOW DRAINER STATUS](#)
- [CHANGE PUMP STATUS](#)
- [CHANGE DRAINER STATUS](#)

11.5.2.100 SHOW PLACEMENT

Warning:

Placement Rules in SQL is an experimental feature. The syntax might change before its GA, and there might also be bugs.

If you understand the risks, you can enable this experiment feature by executing `SET GLOBAL tidb_enable_alter_placement = 1;`.

`SHOW PLACEMENT` summarizes all placement options from direct placement and placement policies, and presents them in canonical form.

11.5.2.100.1 Synopsis

```

ShowStmt ::=  
  "PLACEMENT"

```

11.5.2.100.2 Examples

```

CREATE PLACEMENT POLICY p1 PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us-
    ↪ -west-1" FOLLOWERS=4;
CREATE TABLE t1 (a INT) PLACEMENT POLICY=p1;
CREATE TABLE t2 (a INT) PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us-
    ↪ west-1" FOLLOWERS=4;
SHOW PLACEMENT;

```

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Target	Placement	Scheduling State
POLICY p1	PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us-west-1" ↳ FOLLOWERS=4 NULL	
DATABASE test	PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us-west-1" ↳ FOLLOWERS=4 INPROGRESS	
TABLE test.t1	PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us-west-1" ↳ FOLLOWERS=4 INPROGRESS	
TABLE test.t2	PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us-west-1" ↳ FOLLOWERS=4 INPROGRESS	

4 rows in set (0.00 sec)

11.5.2.100.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.100.4 See also

- [Placement Rules in SQL](#)
- [SHOW PLACEMENT FOR](#)
- [CREATE PLACEMENT POLICY](#)

11.5.2.101 SHOW PLACEMENT FOR

Warning:

Placement Rules in SQL is an experimental feature. The syntax might change before its GA, and there might also be bugs.

If you understand the risks, you can enable this experiment feature by executing `SET GLOBAL tidb_enable_alter_placement = 1;`.

`SHOW PLACEMENT FOR` summarizes all placement options from direct placement and placement policies, and presents them in the canonical form for a specific table, database schema, or partition.

11.5.2.101.1 Synopsis

```
ShowStmt ::=  
    "PLACEMENT" "FOR" ShowPlacementTarget  
  
ShowPlacementTarget ::=  
    DatabaseSym DBName  
  | "TABLE" TableName  
  | "TABLE" TableName "PARTITION" Identifier
```

11.5.2.101.2 Examples

```
CREATE PLACEMENT POLICY p1 PRIMARY_REGION="us-east-1" REGIONS="us-east-1,  
    ↪ -west-1" FOLLOWERS=4;  
use test;  
ALTER DATABASE test PLACEMENT POLICY=p1;  
CREATE TABLE t1 (a INT);  
CREATE TABLE t2 (a INT) PRIMARY_REGION="us-east-1" REGIONS="us-east-1,  
    ↪ west-1" FOLLOWERS=4;  
SHOW PLACEMENT FOR DATABASE test;  
SHOW PLACEMENT FOR TABLE t1;  
SHOW CREATE TABLE t1\G  
SHOW PLACEMENT FOR TABLE t2;  
CREATE TABLE t3 (a INT) PARTITION BY RANGE (a) (PARTITION p1 VALUES LESS  
    ↪ THAN (10), PARTITION p2 VALUES LESS THAN (20) FOLLOWERS=4);  
SHOW PLACEMENT FOR TABLE t3 PARTITION p1;  
SHOW PLACEMENT FOR TABLE t3 PARTITION p2;
```

```
Query OK, 0 rows affected (0.02 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

+-----+
| Target      | Placement          | Scheduling_State |
+-----+
| DATABASE test | PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us-west-1" |
|   ↪ FOLLOWERS=4 | INPROGRESS |
+-----+
| ↪
1 row in set (0.00 sec)

+-----+-----+-----+
| Target      | Placement | Scheduling_State |
+-----+-----+-----+
| TABLE test.t1 | FOLLOWERS=4 | INPROGRESS |
+-----+-----+-----+
1 row in set (0.00 sec)

***** 1. row *****
Table: t1
Create Table: CREATE TABLE `t1` (
  `a` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin /*T! [placement]
  ↪ PLACEMENT POLICY='p1' */
1 row in set (0.00 sec)

+-----+
| Target      | Placement          | Scheduling_State |
+-----+
| ↪
1 row in set (0.00 sec)
```

```

| TABLE test.t2 | PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us-west-1"
  ↳ FOLLOWERS=4 | INPROGRESS |
+-----+
  ↳
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.14 sec)

+-----+
  ↳
| Target           | Placement          | Scheduling_State
  ↳
  ↳ |
+-----+
  ↳
| TABLE test.t3 PARTITION p1 | PRIMARY_REGION="us-east-1" REGIONS="us-east
  ↳ -1,,us-west-1" FOLLOWERS=4 | INPROGRESS |
+-----+
  ↳
1 row in set (0.00 sec)

+-----+-----+-----+
| Target       | Placement | Scheduling_State |
+-----+-----+-----+
| TABLE test.t3 PARTITION p2 | FOLLOWERS=4 | INPROGRESS |
+-----+-----+-----+
1 row in set (0.00 sec)

```

11.5.2.101.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.101.4 See also

- [Placement Rules in SQL](#)
- [SHOW PLACEMENT](#)
- [CREATE PLACEMENT POLICY](#)

11.5.2.102 SHOW PLACEMENT LABELS

Warning:

Placement Rules in SQL is an experimental feature. The syntax might change before its GA, and there might also be bugs.

If you understand the risks, you can enable this experiment feature by executing `SET GLOBAL tidb_enable_alter_placement = 1;`.

`SHOW PLACEMENT LABELS` is used to summarize the labels and values that are available for Placement Rules.

11.5.2.102.1 Synopsis

```
ShowStmt ::=  
    "PLACEMENT" "LABELS"
```

11.5.2.102.2 Examples

```
SHOW PLACEMENT LABELS;
```

```
+-----+-----+  
| Key      | Values          |  
+-----+-----+  
| region   | ["us-east-1"] |  
| zone     | ["us-east-1a"] |  
+-----+-----+  
2 rows in set (0.00 sec)
```

11.5.2.102.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.102.4 See also

- [Placement Rules in SQL](#)
- [SHOW PLACEMENT](#)
- [CREATE PLACEMENT POLICY](#)

11.5.2.103 SHOW PLUGINS

`SHOW PLUGINS` shows all plugins installed in TiDB, including each plugin's status and version information.

11.5.2.103.1 Synopsis

ShowStmt:



Figure 275: ShowStmt

ShowTargetFilterable:

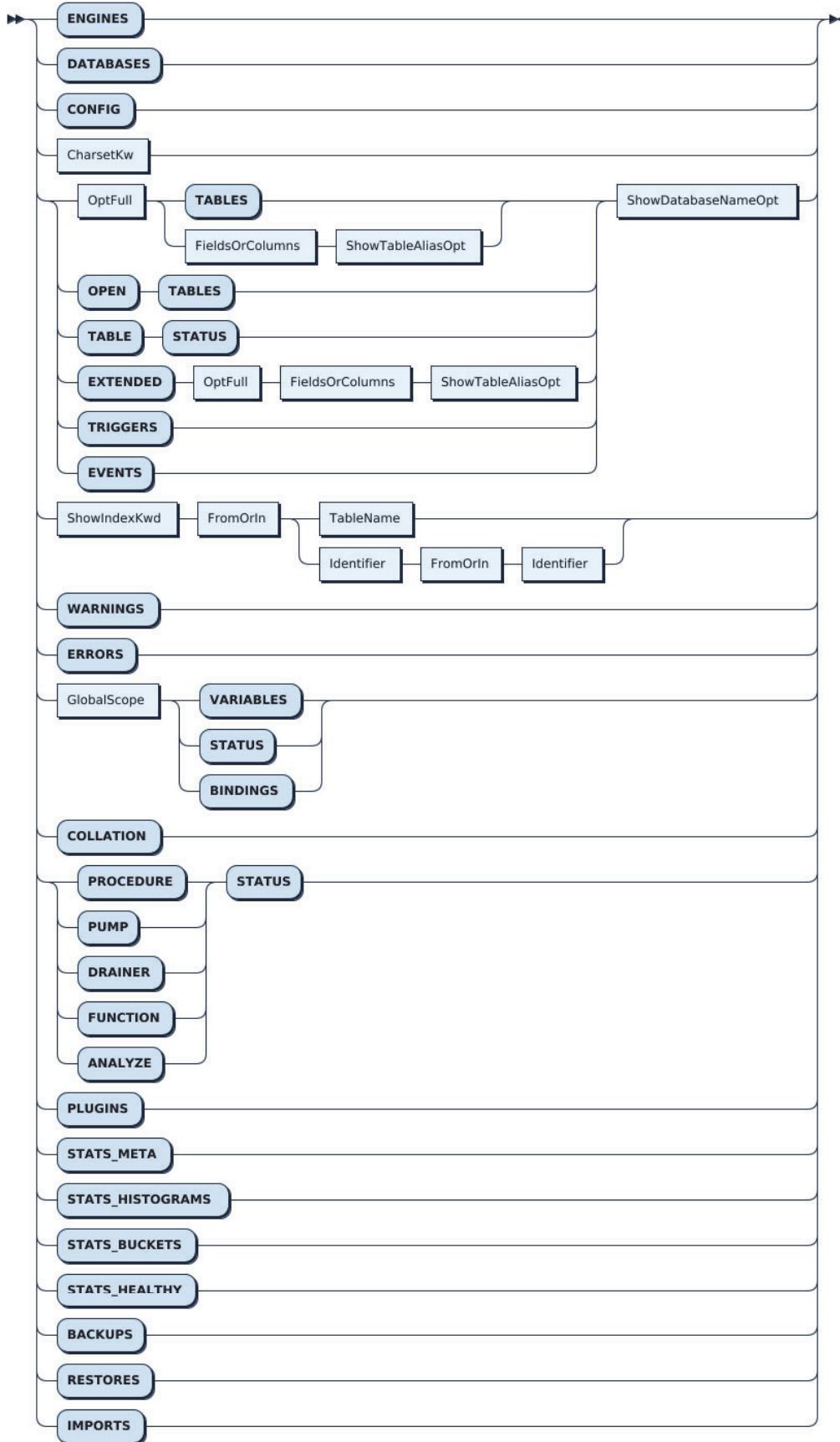


Figure 276: ShowTargetFilterable
1723

11.5.2.103.2 Examples

```
SHOW PLUGINS;
```

Name	Status	Type	Library	License	Version
audit	Ready-enable	Audit	/tmp/tidb/plugin/audit-1.so		1

1 row in set (0.000 sec)

```
SHOW PLUGINS LIKE 'a%';
```

Name	Status	Type	Library	License	Version
audit	Ready-enable	Audit	/tmp/tidb/plugin/audit-1.so		1

1 row in set (0.000 sec)

11.5.2.103.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be [reported via an issue](#) on GitHub.

11.5.2.104 SHOW PRIVILEGES

This statement shows a list of assignable privileges in TiDB. It is a static list, and does not reflect the privileges of the current user.

11.5.2.104.1 Synopsis

ShowStmt:

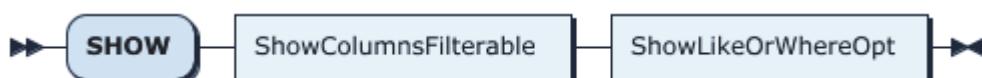


Figure 277: ShowStmt

11.5.2.104.2 Examples

```
mysql> show privileges;
+---+
| Privilege          | Context           | Comment |
+---+
| Alter              | Tables             | To alter the |
|   ↵ table          |                   |           |
| Alter              | Tables             | To alter the |
|   ↵ table          |                   |           |
| Alter routine      | Functions,Procedures | To alter or drop |
|   ↵ stored functions/procedures |           |
| Create              | Databases,Tables,Indexes | To create new |
|   ↵ databases and tables |           |
| Create routine      | Databases          | To use CREATE |
|   ↵ FUNCTION/PROCEDURE |           |
| Create temporary tables | Databases          | To use CREATE |
|   ↵ TEMPORARY TABLE |           |
| Create view         | Tables             | To create new |
|   ↵ views            |           |
| Create user         | Server Admin       | To create new |
|   ↵ users            |           |
| Delete              | Tables             | To delete    |
|   ↵ existing rows    |           |
| Drop                | Databases,Tables   | To drop      |
|   ↵ databases, tables, and views |           |
| Event               | Server Admin       | To create, alter |
|   ↵ , drop and execute events |           |
| Execute              | Functions,Procedures | To execute   |
|   ↵ stored routines   |           |
| File                | File access on server | To read and |
|   ↵ write files on the server |           |
| Grant option         | Databases,Tables,Functions,Procedures | To give to |
|   ↵ other users those privileges you possess |           |
| Index               | Tables             | To create or |
|   ↵ drop indexes     |           |
| Insert               | Tables             | To insert data |
|   ↵ into tables      |           |
| Lock tables          | Databases          | To use LOCK |
|   ↵ TABLES (together with SELECT privilege) |           |
```

Process	Server Admin	To view the
↳ plain text of currently executing queries		
Proxy	Server Admin	To make proxy
↳ user possible		
References	Databases,Tables	To have
↳ references on tables		
Reload	Server Admin	To reload or
↳ refresh tables, logs and privileges		
Replication client	Server Admin	To ask where the
↳ slave or master servers are		
Replication slave	Server Admin	To read binary
↳ log events from the master		
Select	Tables	To retrieve rows
↳ from table		
Show databases	Server Admin	To see all
↳ databases with SHOW DATABASES		
Show view	Tables	To see views
↳ with SHOW CREATE VIEW		
Shutdown	Server Admin	To shut down the
↳ server		
Super	Server Admin	To use KILL
↳ thread, SET GLOBAL, CHANGE MASTER, etc.		
Trigger	Tables	To use triggers
↳		
Create tablespace	Server Admin	To create/alter/
↳ drop tablespaces		
Update	Tables	To update
↳ existing rows		
Usage	Server Admin	No privileges -
↳ allow connect only		
+--		
↳ -----+-----+		
↳		
32 rows in set (0.00 sec)		

11.5.2.104.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.104.4 See also

- SHOW GRANTS
- GRANT <privileges>

11.5.2.105 SHOW [FULL] PROCESSLIST

This statement lists the current sessions connected to the same TiDB server. The Info column contains the query text, which will be truncated unless the optional keyword FULL is specified.

11.5.2.105.1 Synopsis

ShowProcesslistStmt:



Figure 278: ShowProcesslistStmt

OptFull:



Figure 279: OptFull

11.5.2.105.2 Examples

<pre> mysql> SHOW PROCESSLIST; +---+ Id User Host db Command Time State Info +---+ 5 root 127.0.0.1:45970 test Query 0 autocommit SHOW PROCESSLIST +---+ 1 rows in set (0.00 sec) </pre>

11.5.2.105.3 MySQL compatibility

- The **State** column in TiDB is non-descriptive. Representing state as a single value is more complex in TiDB, since queries are executed in parallel and each goroutine will have a different state at any one time.

11.5.2.105.4 See also

- [KILL \[TiDB\]](#)

11.5.2.106 SHOW PROFILES

The `SHOW PROFILES` statement currently only returns an empty result.

11.5.2.106.1 Synopsis

`ShowStmt:`

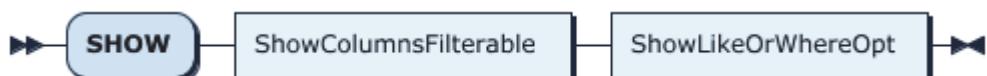


Figure 280: `ShowStmt`

11.5.2.106.2 Examples

```
SHOW PROFILES
```

```
Empty set (0.00 sec)
```

11.5.2.106.3 MySQL compatibility

This statement is included only for compatibility with MySQL. Executing `SHOW PROFILES` always returns an empty result.

11.5.2.107 SHOW PUMP STATUS

The `SHOW PUMP STATUS` statement displays the status information for all Pump nodes in the cluster.

11.5.2.107.1 Examples

```
SHOW PUMP STATUS;
```

NodeID	Address	State	Max_Commit_Ts	Update_Time

```

| pump1 | 127.0.0.1:8250 | Online | 408553768673342237 | 2019-05-01 00:00:01
  ↵   |
+---+
  ↵ -----/-----/-----/-----/-----/
  ↵
| pump2 | 127.0.0.2:8250 | Online | 408553768673342335 | 2019-05-01 00:00:02
  ↵   |
+---+
  ↵ -----/-----/-----/-----/-----/
  ↵
2 rows in set (0.00 sec)

```

11.5.2.107.2 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.107.3 See also

- [SHOW DRAINER STATUS](#)
- [CHANGE PUMP STATUS](#)
- [CHANGE DRAINER STATUS](#)

11.5.2.108 SHOW SCHEMAS

This statement is an alias to [SHOW DATABASES](#). It is included for compatibility with MySQL.

11.5.2.109 SHOW STATS_HEALTHY

The `SHOW STATS_HEALTHY` statement shows an estimation of how accurate statistics are believed to be. Tables with a low percentage health may generate sub-optimal query execution plans.

The health of a table can be improved by running the `ANALYZE` table command. `ANALYZE` runs automatically when the health drops below the `tidb_auto_analyze_ratio` threshold.

11.5.2.109.1 Synopsis

ShowStmt



Figure 281: ShowStmt

ShowTargetFiltertable

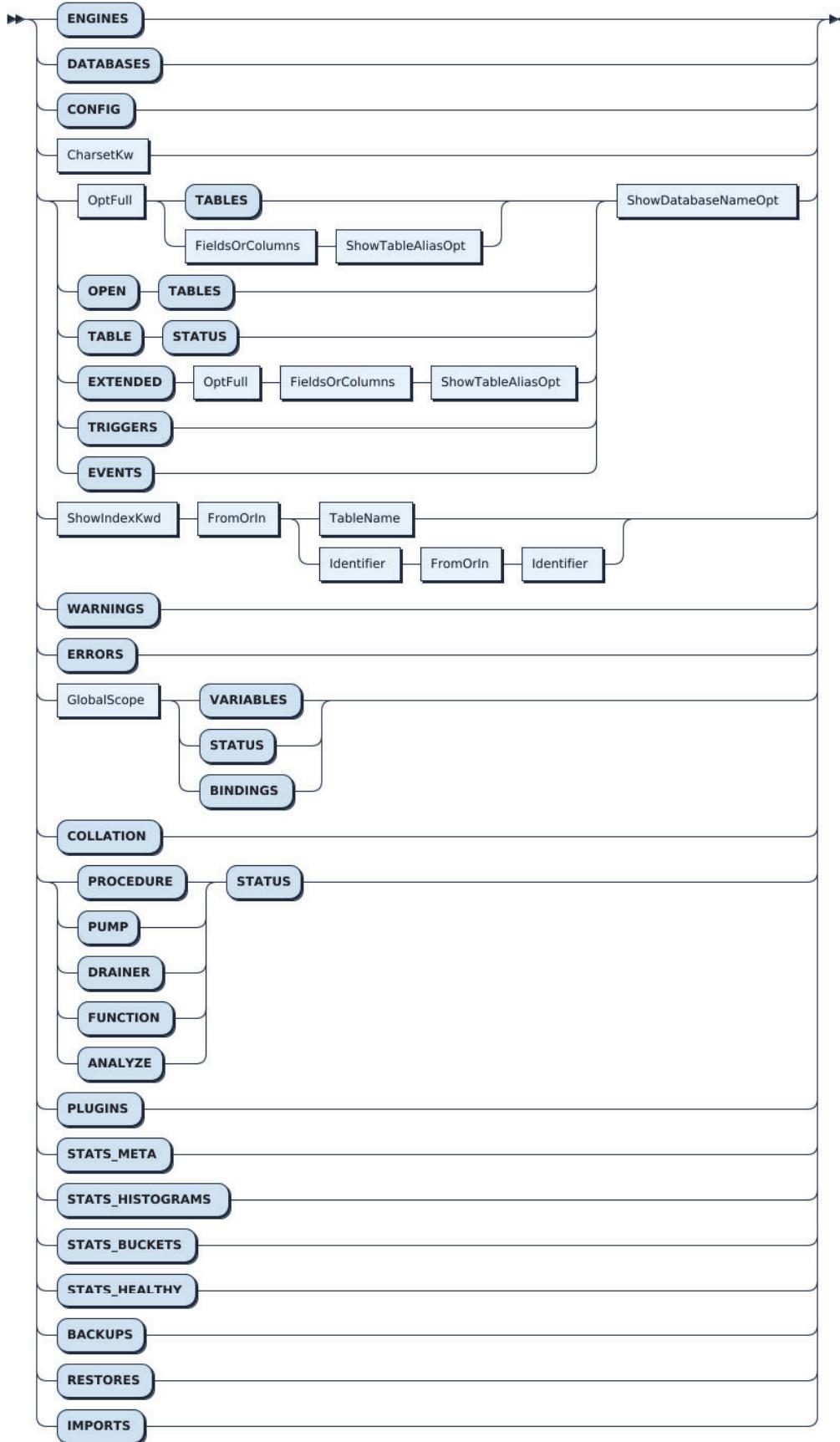


Figure 282: ShowTargetFilterable
1731

ShowLikeOrWhereOpt

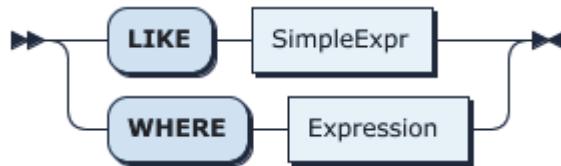


Figure 283: ShowLikeOrWhereOpt

11.5.2.109.2 Examples

Load example data and run ANALYZE:

```

CREATE TABLE t1 (
  id INT NOT NULL PRIMARY KEY auto_increment,
  b INT NOT NULL,
  pad VARBINARY(255),
  INDEX(b)
);

INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255) FROM dual
  ↪ ;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255) FROM t1 a
  ↪ JOIN t1 b JOIN t1 c LIMIT 100000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255) FROM t1 a
  ↪ JOIN t1 b JOIN t1 c LIMIT 100000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255) FROM t1 a
  ↪ JOIN t1 b JOIN t1 c LIMIT 100000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255) FROM t1 a
  ↪ JOIN t1 b JOIN t1 c LIMIT 100000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255) FROM t1 a
  ↪ JOIN t1 b JOIN t1 c LIMIT 100000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(255) FROM t1 a
  ↪ JOIN t1 b JOIN t1 c LIMIT 100000;
SELECT SLEEP(1);
ANALYZE TABLE t1;
SHOW STATS_HEALTHY; # should be 100% healthy
  
```

```

...
mysql> SHOW STATS_HEALTHY;
+-----+-----+-----+-----+
| Db_name | Table_name | Partition_name | Healthy |
+-----+-----+-----+-----+
| test   | t1        |              | 100   |
+-----+-----+-----+-----+
  
```

```
1 row in set (0.00 sec)
```

Perform a bulk update deleting approximately 30% of the records. Check the health of the statistics:

```
DELETE FROM t1 WHERE id BETWEEN 101010 AND 201010; # delete about 30% of
→ records
SHOW STATS_HEALTHY;
```

```
mysql> SHOW STATS_HEALTHY;
+-----+-----+-----+-----+
| Db_name | Table_name | Partition_name | Healthy |
+-----+-----+-----+-----+
| test   | t1        |                 |      50 |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

11.5.2.109.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.109.4 See also

- [ANALYZE](#)
- [Introduction to Statistics](#)

11.5.2.110 SHOW STATS_HISTOGRAMS

This statement shows the histogram information collected by the ANALYZE statement.

11.5.2.110.1 Synopsis

ShowStmt



Figure 284: ShowStmt

ShowTargetFiltertable

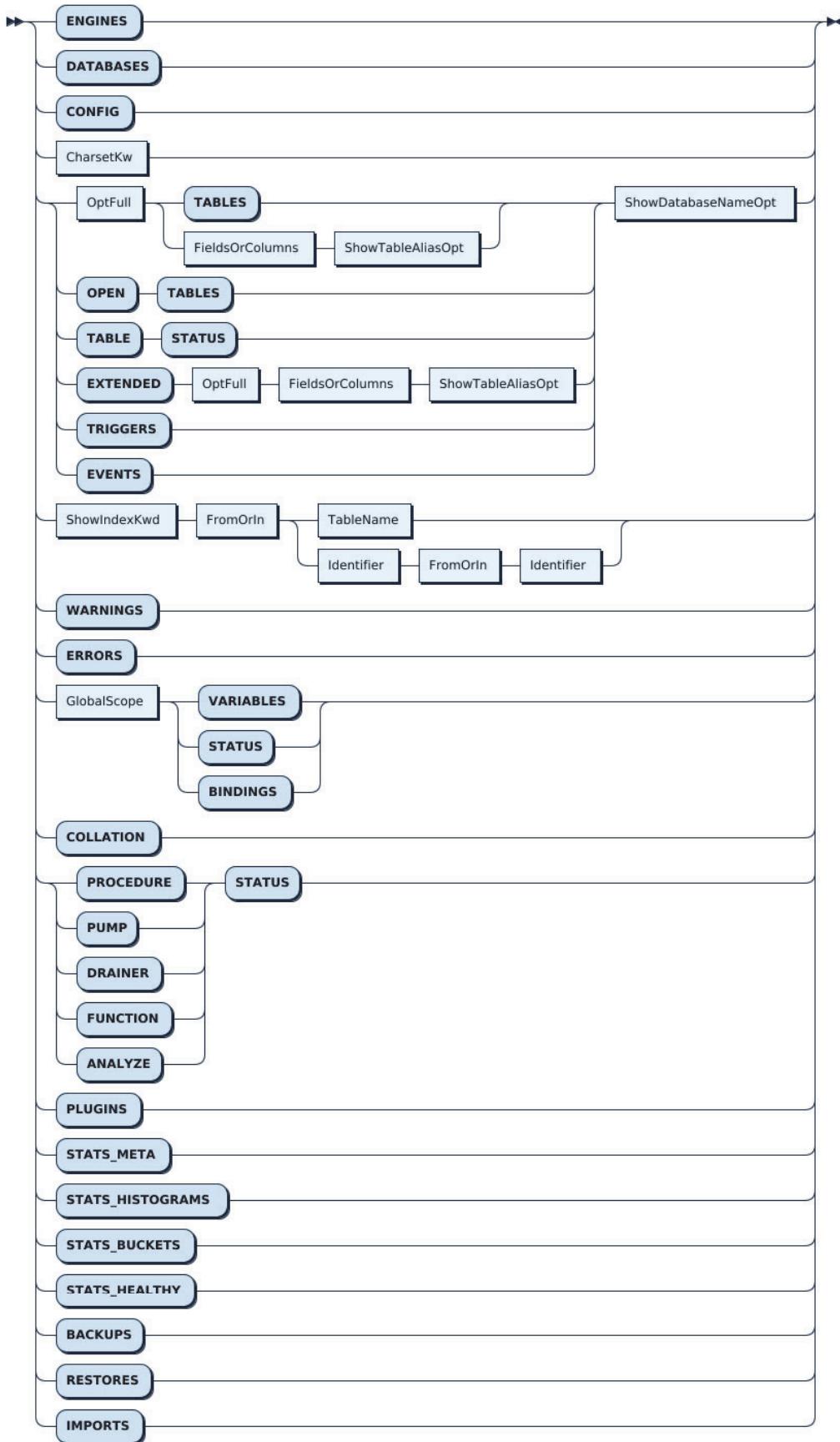


Figure 285: ShowTargetFilterable

ShowLikeOrWhereOpt

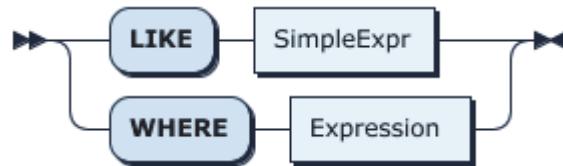


Figure 286: ShowLikeOrWhereOpt

11.5.2.110.2 Examples

```
show stats_histograms;
```

```
+--+
→ -----+-----+-----+-----+
→
| Db_name | Table_name | Partition_name | Column_name | Is_index |
→ Update_time | Distinct_count | Null_count | Avg_col_size |
→ Correlation |
+--+
→ -----+-----+-----+-----+
→
| test | t | | a | | 0 | 2020-05-25
→ 19:20:00 | | 7 | | 0 | | 1 | | 1 |
| test | t2 | | a | | 0 | 2020-05-25
→ 19:20:01 | | 6 | | 0 | | 8 | | 0 |
| test | t2 | | b | | 0 | 2020-05-25
→ 19:20:01 | | 6 | | 0 | | 1.67 | | 1 |
+--+
→ -----+-----+-----+-----+
→
3 rows in set (0.00 sec)
```

```
show stats_histograms where table_name = 't2';
```

```
+--+
→ -----+-----+-----+-----+
→
| Db_name | Table_name | Partition_name | Column_name | Is_index |
→ Update_time | Distinct_count | Null_count | Avg_col_size |
→ Correlation |
+--+
→ -----+-----+-----+-----+
→
```

```
+----+-----+-----+-----+-----+-----+-----+
| test | t2      |       | b      |       | 0   | 2020-05-25 |
|      | ↵ 19:20:01 |       | 0      |       | 1.67 |           |
+----+-----+-----+-----+-----+-----+-----+
| test | t2      |       | a      |       | 0   | 2020-05-25 |
|      | ↵ 19:20:01 |       | 0      |       | 8   |           |
+----+-----+-----+-----+-----+-----+-----+
→-----+-----+-----+-----+-----+-----+
→
2 rows in set (0.00 sec)
```

11.5.2.110.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.110.4 See also

- [ANALYZE](#)
- [Introduction to Statistics](#)

11.5.2.111 SHOW STATS_META

You can use `SHOW STATS_META` to view how many rows are in a table and how many rows are changed in that table. When using this statement, you can filter the needed information by the `ShowLikeOrWhere` clause.

Currently, the `SHOW STATS_META` statement outputs 6 columns:

Syntax element	Description
<code>db_name</code>	Database name
<code>table_name</code>	Table name
<code>partition_name</code>	Partition name
<code>update_time</code>	Last updated time
<code>modify_count</code>	The number of rows modified
<code>row_count</code>	The total row count

注意：

The `update_time` is updated when TiDB updates the `modify_count` and `row_count` fields according to DML statements. So `update_time` is not the last execution time of the `ANALYZE` statement.

11.5.2.111.1 Synopsis

ShowStmt

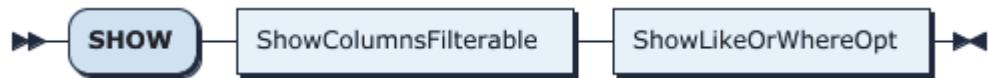


Figure 287: ShowStmt

ShowTargetFiltertable

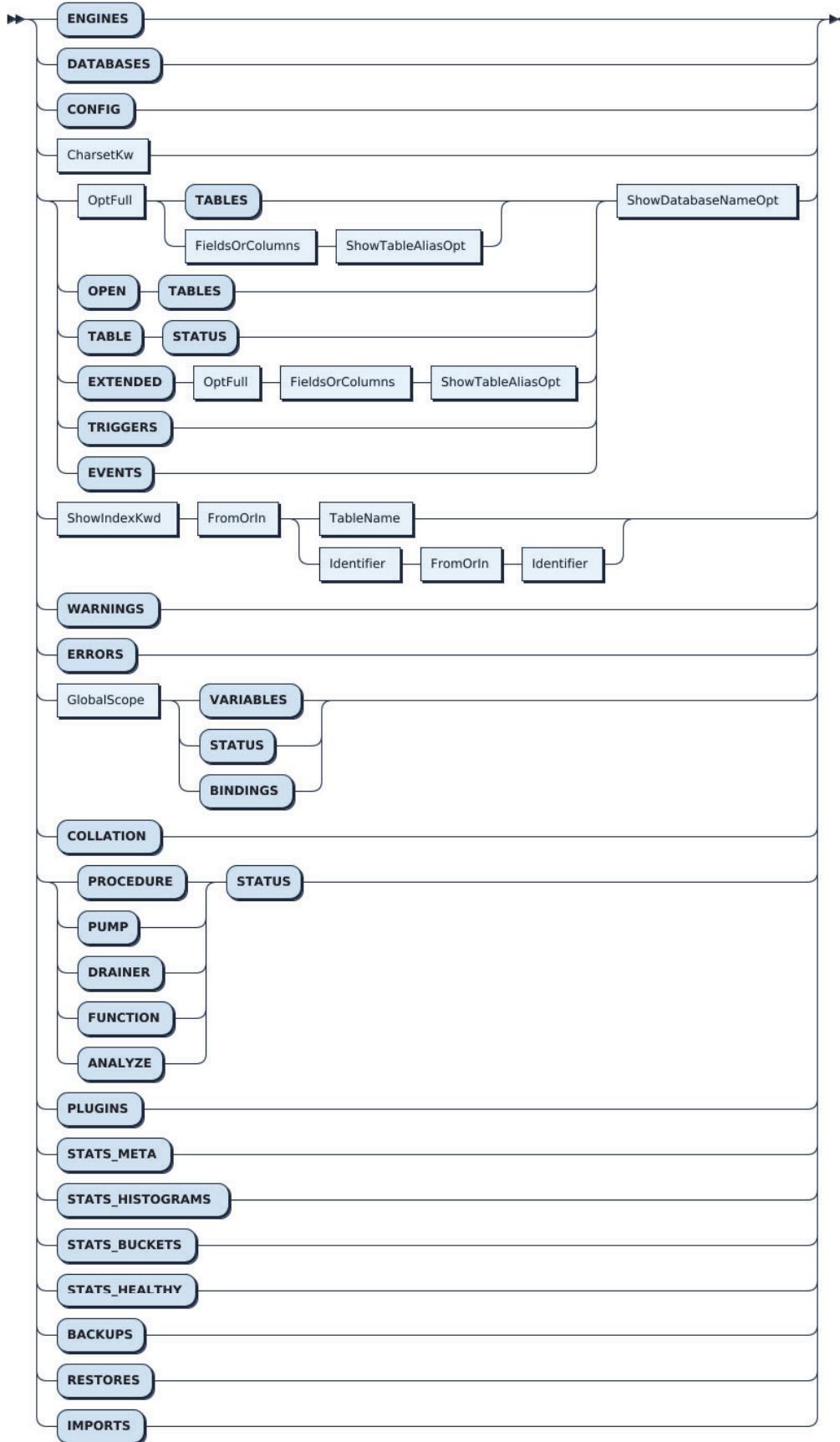


Figure 288: ShowTargetFilterable
1738

ShowLikeOrWhereOpt

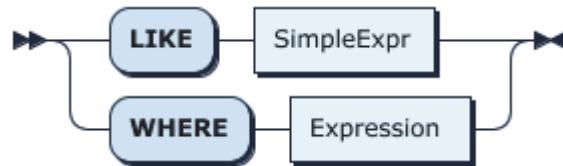


Figure 289: ShowLikeOrWhereOpt

11.5.2.111.2 Examples

```
show stats_meta;
```

Db_name	Table_name	Partition_name	Update_time	Modify_count	Row_count
test	t0		2020-05-15 16:58:00	0	0
test	t1		2020-05-15 16:58:04	0	0
test	t2		2020-05-15 16:58:11	0	0
test	s		2020-05-22 19:46:43	0	0
test	t		2020-05-25 12:04:21	0	0

5 rows in set (0.00 sec)

```
show stats_meta where table_name = 't2';
```

Db_name	Table_name	Partition_name	Update_time	Modify_count	Row_count
test	t2		2020-05-15 16:58:11	0	0

+--	→	-----+-----+-----+-----+	-----+-----+-----+-----+
	→	-----+-----+-----+-----+	-----+-----+-----+-----+
test t2 2020-05-15 16:58:11 0	→	-----+-----+-----+-----+	-----+-----+-----+-----+
0	→	-----+-----+-----+-----+	-----+-----+-----+-----+
+--	→	-----+-----+-----+-----+	-----+-----+-----+-----+
	→	-----+-----+-----+-----+	-----+-----+-----+-----+
1 row in set (0.00 sec)			

11.5.2.111.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.111.4 See also

- [ANALYZE](#)
- [Introduction to Statistics](#)

11.5.2.112 SHOW [GLOBAL|SESSION] STATUS

This statement is included for compatibility with MySQL. It has no effect on TiDB, which uses Prometheus and Grafana for centralized metrics collection instead of SHOW STATUS.

11.5.2.112.1 Synopsis

ShowStmt:



Figure 290: ShowStmt

ShowTargetFilterable:

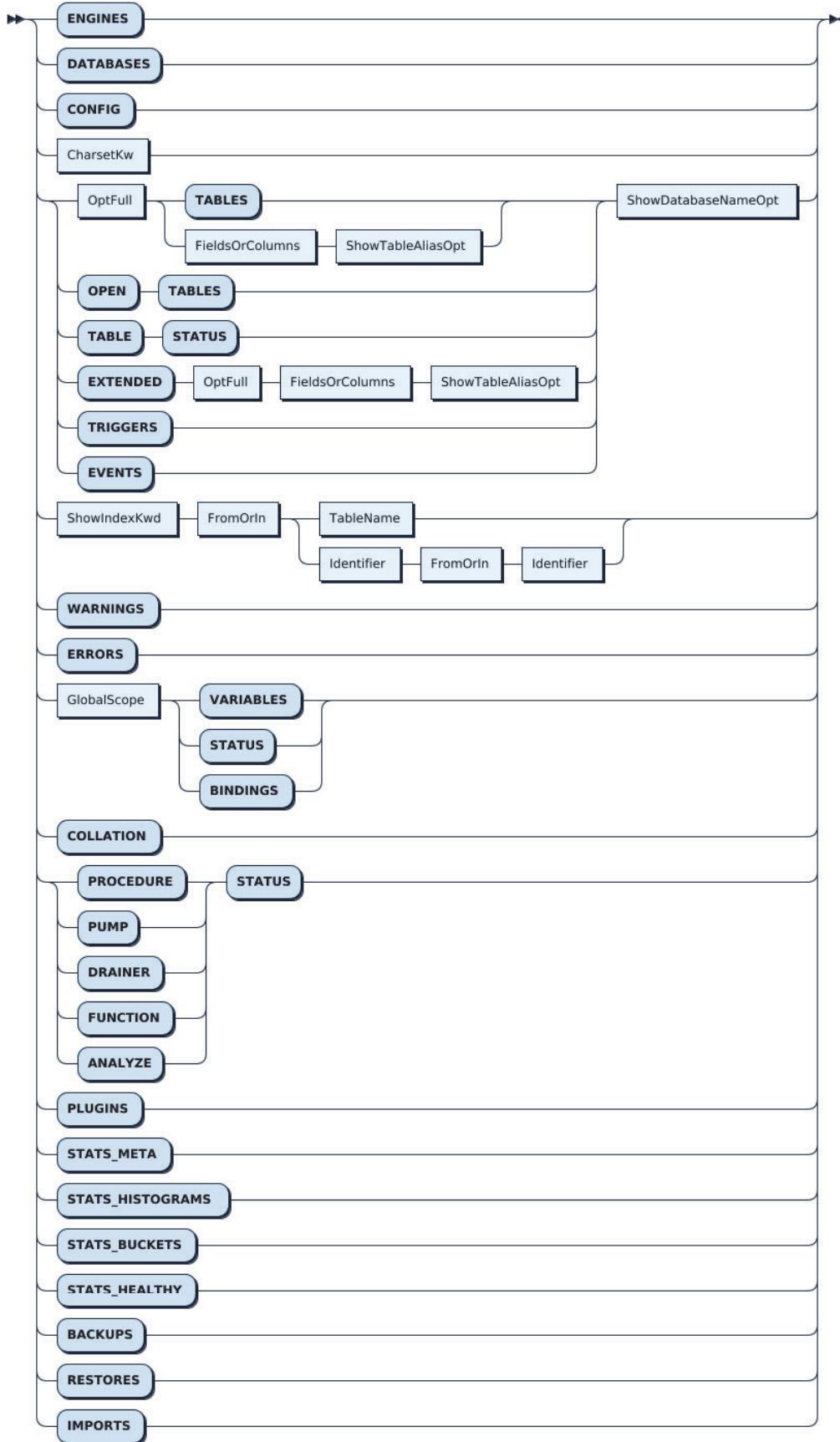


Figure 291: ShowTargetFilterable
1741

GlobalScope:



Figure 292: GlobalScope

11.5.2.112.2 Examples

```

mysql> show status;
+-----+-----+
| Variable_name | Value |
+-----+-----+
| Ssl_cipher_list |          |
| server_id | 93e2e07d-6bb4-4a1b-90b7-e035fae154fe |
| ddl_schema_version | 141 |
| Ssl_verify_mode | 0 |
| Ssl_version |          |
| Ssl_cipher |          |
+-----+
6 rows in set (0.01 sec)

mysql> show global status;
+-----+-----+
| Variable_name | Value |
+-----+-----+
| Ssl_cipher |          |
| Ssl_cipher_list |          |
| Ssl_verify_mode | 0 |
| Ssl_version |          |
| server_id | 93e2e07d-6bb4-4a1b-90b7-e035fae154fe |
| ddl_schema_version | 141 |
+-----+
6 rows in set (0.00 sec)

```

11.5.2.112.3 MySQL compatibility

- This statement is included only for compatibility with MySQL.

11.5.2.112.4 See also

- [FLUSH STATUS](#)

11.5.2.113 SHOW TABLE NEXT_ROW_ID

`SHOW TABLE NEXT_ROW_ID` is used to show the details of some special columns of a table, including:

- `AUTO_INCREMENT` column automatically created by TiDB, namely, `_tidb_rowid` column.
- `AUTO_INCREMENT` column created by users.
- `AUTO_RANDOM` column created by users.
- `SEQUENCE` created by users.

11.5.2.113.1 Synopsis

`ShowTableNextRowIDStmt`:



Figure 293: `ShowTableNextRowIDStmt`

`TableName`:

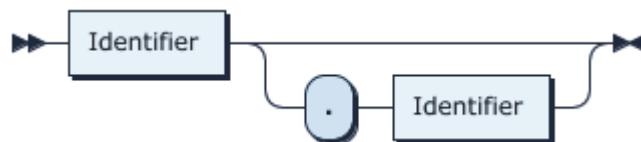


Figure 294: `TableName`

11.5.2.113.2 Examples

For newly created tables, `NEXT_GLOBAL_ROW_ID` is 1 because no Row ID is allocated.

```
create table t(a int);
Query OK, 0 rows affected (0.06 sec)
```

```
show table t next_row_id;
+-----+-----+-----+-----+
| DB_NAME | TABLE_NAME | COLUMN_NAME | NEXT_GLOBAL_ROW_ID |
+-----+-----+-----+-----+
| test   | t          | _tidb_rowid | 1               |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

Data have been written to the table. The TiDB server that inserts the data allocates and caches 30000 IDs at once. Thus, `NEXT_GLOBAL_ROW_ID` is 30001 now.

```
insert into t values (), (), ();
Query OK, 3 rows affected (0.02 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

```
show table t next_row_id;
+-----+
| DB_NAME | TABLE_NAME | COLUMN_NAME | NEXT_GLOBAL_ROW_ID |
+-----+
| test    | t          | _tidb_rowid | 30001   |
+-----+
1 row in set (0.00 sec)
```

11.5.2.113.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.113.4 See also

- [CREATE TABLE](#)
- [AUTO_RANDOM](#)
- [CREATE_SEQUENCE](#)

11.5.2.114 SHOW TABLE REGIONS

The `SHOW TABLE REGIONS` statement is used to show the Region information of a table in TiDB.

11.5.2.114.1 Syntax

```
SHOW TABLE [table_name] REGIONS [WhereClauseOptional];
SHOW TABLE [table_name] INDEX [index_name] REGIONS [WhereClauseOptional];
```

11.5.2.114.2 Synopsis

`ShowTableRegionStmt:`



Figure 295: ShowTableRegionStmt

TableName:

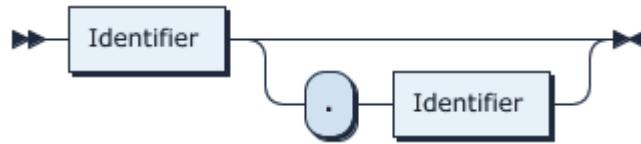


Figure 296: TableName

PartitionNameListOpt:

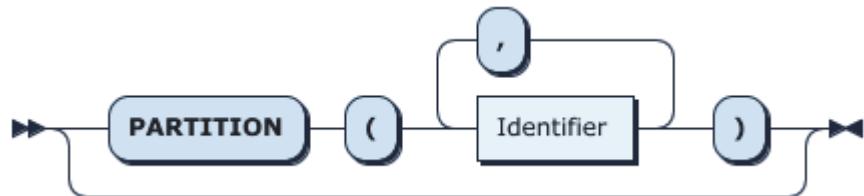


Figure 297: PartitionNameListOpt

WhereClauseOptional:

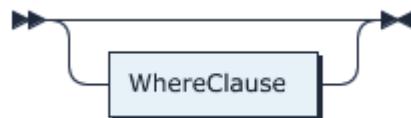


Figure 298: WhereClauseOptional

WhereClause:

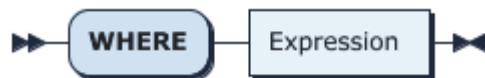


Figure 299: WhereClause

Executing `SHOW TABLE REGIONS` returns the following columns:

- `REGION_ID`: The Region ID.
- `START_KEY`: The start key of the Region.
- `END_KEY`: The end key of the Region.
- `LEADER_ID`: The Leader ID of the Region.
- `LEADER_STORE_ID`: The ID of the store (TiKV) where the Region leader is located.
- `PEERS`: The IDs of all Region replicas.
- `SCATTERING`: Whether the Region is being scheduled. 1 means true.

- WRITTEN_BYTES: The estimated amount of data written into the Region within one heartbeat cycle. The unit is byte.
- READ_BYTES: The estimated amount of data read from the Region within one heartbeat cycle. The unit is byte.
- APPROXIMATE_SIZE(MB): The estimated amount of data in the Region. The unit is megabytes (MB).
- APPROXIMATE_KEYS: The estimated number of Keys in the Region.

Note:

The values of WRITTEN_BYTES, READ_BYTES, APPROXIMATE_SIZE(MB), APPROXIMATE_KEYS are not accurate data. They are estimated data from PD based on the heartbeat information that PD receives from the Region.

11.5.2.114.3 Examples

Create an example table with enough data that fills a few Regions:

```
CREATE TABLE t1 (
    id INT NOT NULL PRIMARY KEY auto_increment,
    b INT NOT NULL,
    pad1 VARBINARY(1024),
    pad2 VARBINARY(1024),
    pad3 VARBINARY(1024)
);
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(1024),
    ↪ RANDOM_BYTES(1024), RANDOM_BYTES(1024) FROM dual;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(1024),
    ↪ RANDOM_BYTES(1024), RANDOM_BYTES(1024) FROM t1 a JOIN t1 b JOIN t1 c
    ↪ LIMIT 10000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(1024),
    ↪ RANDOM_BYTES(1024), RANDOM_BYTES(1024) FROM t1 a JOIN t1 b JOIN t1 c
    ↪ LIMIT 10000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(1024),
    ↪ RANDOM_BYTES(1024), RANDOM_BYTES(1024) FROM t1 a JOIN t1 b JOIN t1 c
    ↪ LIMIT 10000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(1024),
    ↪ RANDOM_BYTES(1024), RANDOM_BYTES(1024) FROM t1 a JOIN t1 b JOIN t1 c
    ↪ LIMIT 10000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(1024),
    ↪ RANDOM_BYTES(1024), RANDOM_BYTES(1024) FROM t1 a JOIN t1 b JOIN t1 c
    ↪ LIMIT 10000;
```

```

INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(1024),
    ↪ RANDOM_BYTES(1024), RANDOM_BYTES(1024) FROM t1 a JOIN t1 b JOIN t1 c
    ↪ LIMIT 10000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(1024),
    ↪ RANDOM_BYTES(1024), RANDOM_BYTES(1024) FROM t1 a JOIN t1 b JOIN t1 c
    ↪ LIMIT 10000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(1024),
    ↪ RANDOM_BYTES(1024), RANDOM_BYTES(1024) FROM t1 a JOIN t1 b JOIN t1 c
    ↪ LIMIT 10000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(1024),
    ↪ RANDOM_BYTES(1024), RANDOM_BYTES(1024) FROM t1 a JOIN t1 b JOIN t1 c
    ↪ LIMIT 10000;
INSERT INTO t1 SELECT NULL, FLOOR(RAND()*1000), RANDOM_BYTES(1024),
    ↪ RANDOM_BYTES(1024), RANDOM_BYTES(1024) FROM t1 a JOIN t1 b JOIN t1 c
    ↪ LIMIT 10000;
SELECT SLEEP(5);
SHOW TABLE t1 REGIONS;

```

The output should show that the table is split into Regions. The REGION_ID, START_KEY and END_KEY may not match exactly:

```

...
mysql> SHOW TABLE t1 REGIONS;
+---+
| REGION_ID | START_KEY | END_KEY      | LEADER_ID | LEADER_STORE_ID | PEERS |
|           | SCATTERING | WRITTEN_BYTES | READ_BYTES | APPROXIMATE_SIZE(MB) | APPROXIMATE_KEYS |
+---+
| 94 | t_75_          | t_75_r_31717 | 95 | 1 | 95 |
|     | 0 | 0 | 0 | 112 |
| 207465 |
| 96 | t_75_r_31717 | t_75_r_63434 | 97 | 1 | 97 |
|     | 0 | 0 | 0 | 97 |
| 0 |
| 2 | t_75_r_63434 |           | 3 | 1 | 3 |
|     | 0 | 269323514 | 66346110 | 245 |
| 162020 |
+---+

```

```
+-----+-----+-----+-----+
|       |
|       |
3 rows in set (0.00 sec)
```

In the output above, a START_KEY of t_75_r_31717 and END_KEY of t_75_r_63434 shows that data with a PRIMARY KEY between 31717 and 63434 is stored in this Region. The prefix t_75_ indicates that this is the Region for a table (t) which has an internal table ID of 75. An empty key value for START_KEY or END_KEY indicates negative infinity or positive infinity respectively.

TiDB automatically rebalances Regions as needed. For manual rebalancing, use the SPLIT TABLE REGION statement:

```
mysql> SPLIT TABLE t1 BETWEEN (31717) AND (63434) REGIONS 2;
```

```
+-----+-----+
| TOTAL_SPLIT_REGION | SCATTER_FINISH_RATIO |
+-----+-----+
|           1 |                 1 |
+-----+-----+
1 row in set (42.34 sec)
```

```
mysql> SHOW TABLE t1 REGIONS;
```

```
+-----+-----+-----+-----+-----+-----+-----+
|       |
|       |
| REGION_ID | START_KEY | END_KEY      | LEADER_ID | LEADER_STORE_ID | PEERS |
```

SCATTERING	WRITTEN_BYTES	READ_BYTES	APPROXIMATE_SIZE(MB)	APPROXIMATE_KEYS		
94	t_75_	t_75_r_31717	95	1	95	
0	0	0	112	207465		
98	t_75_r_31717	t_75_r_47575	99	1	99	
0	1325	0	53	12052		
96	t_75_r_47575	t_75_r_63434	97	1	97	
0	1526	0	48	0		
2	t_75_r_63434		3	1	3	
0	0	55752049	60	0		

```
4 rows in set (0.00 sec)
```

The above output shows that Region 96 was split, with a new Region 98 being created. The remaining Regions in the table were unaffected by the split operation. This is confirmed by the output statistics:

- TOTAL_SPLIT_REGION indicates the number of newly split Regions. In this example, the number is 1.
- SCATTER_FINISH_RATIO indicates the rate at which the newly split Regions are successfully scattered. 1.0 means that all Regions are scattered.

For a more detailed example:

REGION_ID	START_KEY	END_KEY	LEADER_ID	LEADER_STORE_ID	PEERS
	SCATTERING	WRITTEN_BYTES	READ_BYTES	APPROXIMATE_SIZE(MB)	APPROXIMATE_KEYS
102	t_43_r	t_43_r_20000	118	7	105, 118,
119	0	0	0	1	0
106	t_43_r_20000	t_43_r_40000	120	7	107, 108,
120	0	23	0	1	0
110	t_43_r_40000	t_43_r_60000	112	9	112, 113,
121	0	0	0	1	0
114	t_43_r_60000	t_43_r_80000	122	7	115, 122,
123	0	35	0	1	0
3	t_43_r_80000		93	8	5, 73, 93
	0	0	0	1	0
98	t_43_	t_43_r	99	1	99, 100,
101	0	0	0	1	0

```
6 rows in set
```

In the above example:

- Table t corresponds to six Regions. In these Regions, 102, 106, 110, 114, and 3 store the row data and 98 stores the index data.
- For START_KEY and END_KEY of Region 102, t_43 indicates the table prefix and ID. _r is the prefix of the record data in table t. _i is the prefix of the index data.
- In Region 102, START_KEY and END_KEY mean that record data in the range of [-inf, ↳ 20000) is stored. In similar way, the ranges of data storage in Regions (106, 110, 114, 3) can also be calculated.
- Region 98 stores the index data. The start key of table t's index data is t_43_i, which is in the range of Region 98.

To check the Region that corresponds to table t in store 1, use the WHERE clause:

test> show table t regions where leader_store_id =1;						
REGION_ID	START_KEY	END_KEY	LEADER_ID	LEADER_STORE_ID	PEERS	
98	t_43_	t_43_r	99	1	99, 100, 101	0
	0	0	1		0	

Use SPLIT TABLE REGION to split the index data into Regions. In the following example, the index data name of table t is split into two Regions in the range of [a,z].

test> split table t index name between ("a") and ("z") regions 2;	
TOTAL_SPLIT_REGION	SCATTER_FINISH_RATIO
2	1.0
1 row in set	

Now table t corresponds to seven Regions. Five of them (102, 106, 110, 114, 3) store the record data of table t and another two (135, 98) store the index data name.

test> show table t regions;

REGION_ID	START_KEY	END_KEY	LEADER_ID		
	LEADER_STORE_ID	PEERS	SCATTERING	WRITTEN_BYTES	READ_BYTES
	APPROXIMATE_SIZE(MB)	APPROXIMATE_KEYS			
<hr/>					
102	t_43_r	t_43_r_20000	118		
7		105, 118, 119 0 0 0 1			
		0			
106	t_43_r_20000	t_43_r_40000	120		
7		108, 120, 126 0 0 0 1			
		0			
110	t_43_r_40000	t_43_r_60000	112		
9		112, 113, 121 0 0 0 1			
		0			
114	t_43_r_60000	t_43_r_80000	122		
7		115, 122, 123 0 35 0 1			
		0			
3	t_43_r_80000		93		
8		73, 93, 128 0 0 0 1			
		0			
135	t_43_i_1_	t_43_i_1_016d8000000000000000	139		
2		138, 139, 140 0 35 0 1			
		0			
98	t_43_i_1_016d8000000000000000	t_43_r	99		
1		99, 100, 101 0 0 0 1			
		0			
<hr/>					
7 rows in set					

11.5.2.114.4 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.114.5 See also

- [SPLIT REGION](#)
- [CREATE TABLE](#)

11.5.2.115 SHOW TABLE STATUS

This statement shows various statistics about tables in TiDB. If the statistics appear out of date, it is recommended to run `ANALYZE TABLE`.

11.5.2.115.1 Synopsis

`ShowTableStatusStmt`:

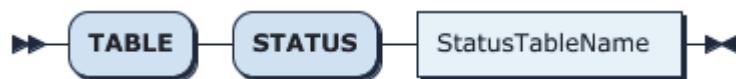


Figure 300: `ShowTableStatusStmt`

`FromOrIn`:

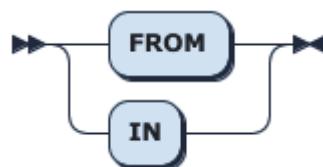


Figure 301: `FromOrIn`

`StatusTableName`:



Figure 302: `StatusTableName`

11.5.2.115.2 Examples

```

mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT
   ↪ NOT NULL);
Query OK, 0 rows affected (0.11 sec)

mysql> INSERT INTO t1 (c1) VALUES (1),(2),(3),(4),(5);
Query OK, 5 rows affected (0.02 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> SHOW TABLE STATUS LIKE 't1'\G
***** 1. row *****
  Name: t1
  Engine: InnoDB
  Version: 10
  Row_format: Compact

```

```

      Rows: 0
Avg_row_length: 0
  Data_length: 0
Max_data_length: 0
  Index_length: 0
    Data_free: 0
Auto_increment: 30001
  Create_time: 2019-04-19 08:32:06
  Update_time: NULL
  Check_time: NULL
  Collation: utf8mb4_bin
  Checksum:
Create_options:
  Comment:
1 row in set (0.00 sec)

mysql> analyze table t1;
Query OK, 0 rows affected (0.12 sec)

mysql> SHOW TABLE STATUS LIKE 't1'\G
***** 1. row *****
      Name: t1
      Engine: InnoDB
      Version: 10
      Row_format: Compact
      Rows: 5
Avg_row_length: 16
  Data_length: 80
Max_data_length: 0
  Index_length: 0
    Data_free: 0
Auto_increment: 30001
  Create_time: 2019-04-19 08:32:06
  Update_time: NULL
  Check_time: NULL
  Collation: utf8mb4_bin
  Checksum:
Create_options:
  Comment:
1 row in set (0.00 sec)

```

11.5.2.115.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility

differences should be reported via an issue on GitHub.

11.5.2.115.4 See also

- SHOW TABLES
- CREATE TABLE
- DROP TABLE
- SHOW CREATE TABLE

11.5.2.116 SHOW [FULL] TABLES

This statement shows a list of tables and views in the currently selected database. The optional keyword FULL indicates if a table is of type BASE TABLE or VIEW.

To show tables in a different database, use `SHOW TABLES IN DatabaseName`.

11.5.2.116.1 Synopsis

`ShowTablesStmt:`



Figure 303: ShowTablesStmt

`OptFull:`

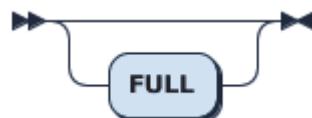


Figure 304: OptFull

`ShowDatabaseNameOpt:`



Figure 305: ShowDatabaseNameOpt

`ShowLikeOrWhereOpt:`

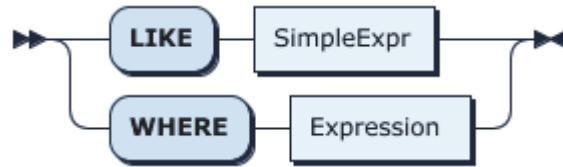


Figure 306: ShowLikeOrWhereOpt

11.5.2.116.2 Examples

```
mysql> CREATE TABLE t1 (a int);
Query OK, 0 rows affected (0.12 sec)
```

```
mysql> CREATE VIEW v1 AS SELECT 1;
Query OK, 0 rows affected (0.10 sec)
```

```
mysql> SHOW TABLES;
+-----+
| Tables_in_test |
+-----+
| t1           |
| v1           |
+-----+
2 rows in set (0.00 sec)
```

```
mysql> SHOW FULL TABLES;
+-----+-----+
| Tables_in_test | Table_type |
+-----+-----+
| t1            | BASE TABLE |
| v1            | VIEW       |
+-----+-----+
2 rows in set (0.00 sec)
```

```
mysql> SHOW TABLES IN mysql;
+-----+
| Tables_in_mysql   |
+-----+
| GLOBAL_VARIABLES |
| bind_info         |
| columns_priv     |
| db                |
| default_roles    |
| expr_pushdown_blacklist |
| gc_delete_range  |
| gc_delete_range_done |
+-----+
```

```

| global_priv      |
| help_topic       |
| opt_rule_blacklist |
| role_edges       |
| stats_buckets    |
| stats_feedback   |
| stats_histograms |
| stats_meta       |
| stats_top_n      |
| tables_priv      |
| tidb              |
| user              |
+-----+
20 rows in set (0.00 sec)

```

11.5.2.116.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.116.4 See also

- [CREATE TABLE](#)
- [DROP TABLE](#)
- [SHOW CREATE TABLE](#)

11.5.2.117 SHOW [GLOBAL|SESSION] VARIABLES

This statement shows a list of variables for the scope of either GLOBAL or SESSION. If no scope is specified, the default scope of SESSION will apply.

11.5.2.117.1 Synopsis

ShowStmt:



Figure 307: ShowStmt

ShowTargetFilterable:

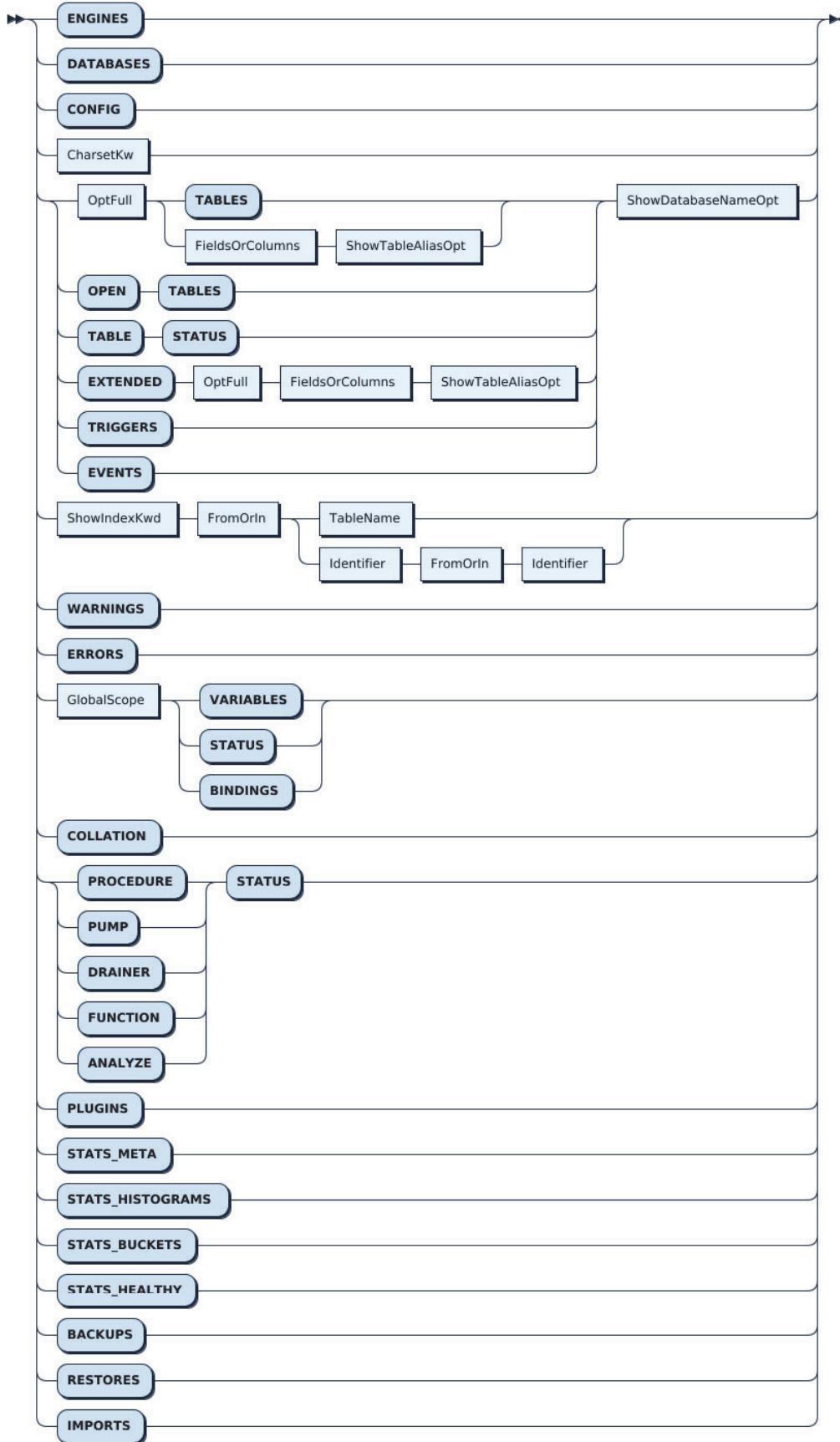


Figure 308: ShowTargetFilterable
1757

GlobalScope:

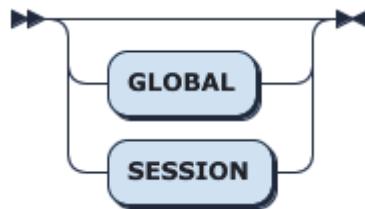


Figure 309: GlobalScope

11.5.2.117.2 Examples

List all TiDB specific variables. For detailed description, refer to [System Variables](#).

Variable_name	Value
tidb_allow_batch_cop	0
tidb_allow_remove_auto_inc	0
tidb_auto_analyze_end_time	23:59 +0000
tidb_auto_analyze_ratio	0.5
tidb_auto_analyze_start_time	00:00 +0000
tidb_backoff_lock_fast	100
tidb_backoff_weight	2
tidb_batch_commit	0
tidb_batch_delete	0
tidb_batch_insert	0
tidb_build_stats_concurrency	4
tidb_capture_plan_baselines	off
tidb_check_mb4_value_in_utf8	1
tidb_checksum_table_concurrency	4
tidb_config	
tidb_constraint_check_in_place	0
tidb_current_ts	0
tidb_ddl_error_count_limit	512
tidb_ddl_reorg_batch_size	256
tidb_ddl_reorg_priority	PRIORITY_LOW
tidb_ddl_reorg_worker_cnt	4
tidb_disable_txn_auto_retry	1
tidb_distsql_scan_concurrency	15
tidb_dml_batch_size	20000
tidb_enable_cascades_planner	0
tidb_enable_chunk_rpc	1
tidb_enable_collect_execution_info	1

tidb_enable_fast_analyze	0	
tidb_enable_index_merge	0	
tidb_enable_noop_functions	0	
tidb_enable_radix_join	0	
tidb_enable_slow_log	1	
tidb_enable_stmt_summary	1	
tidb_enable_table_partition	on	
tidb_enable_vectorized_expression	1	
tidb_enable_window_function	1	
tidb_evolve_plan_baselines	off	
tidb_evolve_plan_task_end_time	23:59 +0000	
tidb_evolve_plan_task_max_time	600	
tidb_evolve_plan_task_start_time	00:00 +0000	
tidb_expensive_query_time_threshold	60	
tidb_force_priority	NO_PRIORITY	
tidb_general_log	0	
tidb_hash_join_concurrency	5	
tidb_hashagg_final_concurrency	4	
tidb_hashagg_partial_concurrency	4	
tidb_index_join_batch_size	25000	
tidb_index_lookup_concurrency	4	
tidb_index_lookup_join_concurrency	4	
tidb_index_lookup_size	20000	
tidb_index_serial_scan_concurrency	1	
tidb_init_chunk_size	32	
tidb_isolation_read_engines	tikv, tiflash, tidb	
tidb_low_resolution_tso	0	
tidb_max_chunk_size	1024	
tidb_max_delta_schema_count	1024	
tidb_mem_quota_hashjoin	34359738368	
tidb_mem_quota_indexlookupjoin	34359738368	
tidb_mem_quota_indexlookupreader	34359738368	
tidb_mem_quota_mergejoin	34359738368	
tidb_mem_quota_nestedloopapply	34359738368	
tidb_mem_quota_query	1073741824	
tidb_mem_quota_sort	34359738368	
tidb_mem_quota_topn	34359738368	
tidb_metric_query_range_duration	60	
tidb_metric_query_step	60	
tidb_opt_agg_push_down	0	
tidb_opt_concurrency_factor	3	
tidb_opt_copcpu_factor	3	
tidb_opt_correlation_exp_factor	1	
tidb_opt_correlation_threshold	0.9	
tidb_opt_cpu_factor	3	

```

| tidb_opt_desc_factor | 3
| tidb_opt_disk_factor | 1.5
| tidb_opt_distinct_agg_push_down | 0
| tidb_opt_insubq_to_join_and_agg | 1
| tidb_opt_join_reorder_threshold | 0
| tidb_opt_memory_factor | 0.001
| tidb_opt_network_factor | 1
| tidb_opt_scan_factor | 1.5
| tidb_opt_seek_factor | 20
| tidb_opt_write_row_id | 0
| tidb_optimizer_selectivity_level | 0
| tidb_pprof_sql_cpu | 0
| tidb_projectionConcurrency | 4
| tidb_query_log_max_len | 4096
| tidb_record_plan_in_slow_log | 1
| tidb_replica_read | leader
| tidb_retry_limit | 10
| tidb_row_format_version | 2
| tidb_scatter_region | 0
| tidb_skip_isolation_level_check | 0
| tidb_skip_utf8_check | 0
| tidb_slow_log_threshold | 300
| tidb_slow_query_file | tidb-slow.log
| tidb_snapshot |
| tidb_stmt_summary_history_size | 24
| tidb_stmt_summary_internal_query | 0
| tidb_stmt_summary_max_sql_length | 4096
| tidb_stmt_summary_max_stmt_count | 3000
| tidb_stmt_summary_refresh_interval | 1800
| tidb_store_limit | 0
| tidb_txn_mode |
| tidb_use_plan_baselines | on
| tidb_wait_split_region_finish | 1
| tidb_wait_split_region_timeout | 300
| tidb_window_concurrency | 4
+-----+
108 rows in set (0.01 sec)

mysql> SHOW GLOBAL VARIABLES LIKE 'time_zone%';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| time_zone | SYSTEM |
+-----+-----+
1 row in set (0.00 sec)

```

11.5.2.117.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.117.4 See also

- [SET \[GLOBAL|SESSION\]](#)

11.5.2.118 SHOW WARNINGS

This statement shows a list of warnings that occurred for previously executed statements in the current client connection. As in MySQL, the `sql_mode` impacts which statements will cause errors vs. warnings considerably.

11.5.2.118.1 Synopsis

`ShowWarningsStmt:`



Figure 310: ShowWarningsStmt

11.5.2.118.2 Examples

```

mysql> CREATE TABLE t1 (a INT UNSIGNED);
Query OK, 0 rows affected (0.11 sec)

mysql> INSERT INTO t1 VALUES (0);
Query OK, 1 row affected (0.02 sec)

mysql> SELECT 1/a FROM t1;
+-----+
| 1/a  |
+-----+
| NULL |
+-----+
1 row in set, 1 warning (0.00 sec)

mysql> SHOW WARNINGS;
+-----+-----+-----+
| Level | Code | Message      |
+-----+-----+-----+

```

```
+-----+-----+
| Warning | 1365 | Division by 0 |
+-----+-----+
1 row in set (0.00 sec)

mysql> INSERT INTO t1 VALUES (-1);
ERROR 1264 (22003): Out of range value for column 'a' at row 1
mysql> SELECT * FROM t1;
+---+
| a |
+---+
| 0 |
+---+
1 row in set (0.00 sec)

mysql> SET sql_mode='';
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO t1 VALUES (-1);
Query OK, 1 row affected, 1 warning (0.01 sec)

mysql> SHOW WARNINGS;
+-----+-----+
| Level | Code | Message           |
+-----+-----+
| Warning | 1690 | constant -1 overflows int |
+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT * FROM t1;
+---+
| a |
+---+
| 0 |
| 0 |
+---+
2 rows in set (0.00 sec)
```

11.5.2.118.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be [reported via an issue on GitHub](#).

11.5.2.118.4 See also

- SHOW ERRORS

11.5.2.119 SHUTDOWN

The `SHUTDOWN` statement is used to perform a shutdown operation in TiDB. Execution of the `SHUTDOWN` statement requires the user to have `SHUTDOWN` privilege.

11.5.2.119.1 Synopsis

Statement:



Figure 311: Statement

11.5.2.119.2 Examples

```
SHUTDOWN;
```

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

11.5.2.119.3 MySQL compatibility

Note:

Because TiDB is a distributed database, the shutdown operation in TiDB stops the client-connected TiDB instance, not the entire TiDB cluster.

The `SHUTDOWN` statement is partly compatible with MySQL. If you encounter any compatibility issues, you can report the issues [on GitHub](#).

11.5.2.120 Split Region

For each new table created in TiDB, one `Region` is segmented by default to store the data of this table. This default behavior is controlled by `split-table` in the TiDB configuration file. When the data in this Region exceeds the default Region size limit, the Region starts to split into two.

In the above case, because there is only one Region at the beginning, all write requests occur on the TiKV where the Region is located. If there are a large number of writes for the newly created table, hotspots are caused.

To solve the hotspot problem in the above scenario, TiDB introduces the pre-split function, which can pre-split multiple Regions for a certain table according to the specified parameters and scatter them to each TiKV node.

11.5.2.120.1 Synopsis

SplitRegionStmt:

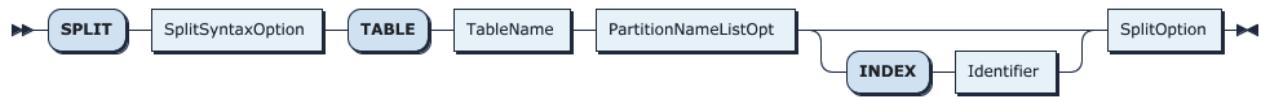


Figure 312: SplitRegionStmt

SplitSyntaxOption:



Figure 313: SplitSyntaxOption

TableName:

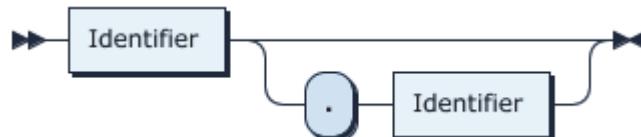


Figure 314: TableName

PartitionNameListOpt:

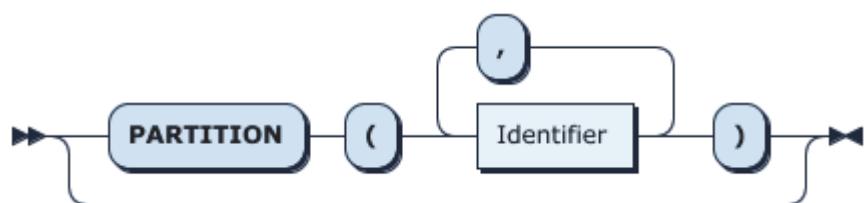


Figure 315: PartitionNameListOpt

SplitOption:



Figure 316: SplitOption

RowValue:



Figure 317: RowValue

Int64Num:



Figure 318: Int64Num

11.5.2.120.2 Usage of Split Region

There are two types of Split Region syntax:

- The syntax of even split:

```
SPLIT TABLE table_name [INDEX index_name] BETWEEN (lower_value) AND (
    ↪ upper_value) REGIONS region_num
```

BETWEEN lower_value AND upper_value REGIONS region_num defines the upper boundary, the lower boundary, and the Region amount. Then the current region will be evenly spilt into the number of regions (as specified in region_num) between the upper and lower boundaries.

- The syntax of uneven split:

```
SPLIT TABLE table_name [INDEX index_name] BY (value_list) [, (
    ↪ value_list)] ...
```

BY value_list... specifies a series of points manually, based on which the current Region is spilt. It is suitable for scenarios with unevenly distributed data.

The following example shows the result of the `SPLIT` statement:

TOTAL_SPLIT_REGION	SCATTER_FINISH_RATIO
4	1.0

- `TOTAL_SPLIT_REGION`: the number of newly split Regions.
- `SCATTER_FINISH_RATIO`: the completion rate of scattering for newly split Regions. 1.0 means that all Regions are scattered. 0.5 means that only half of the Regions are scattered and the rest are being scattered.

Note:

The following two session variables might affect the behavior of the `SPLIT` statement:

- `tidb_wait_split_region_finish`: It might take a while to scatter the Regions. This duration depends on PD scheduling and TiKV loads. This variable is used to control when executing the `SPLIT REGION` statement whether to return the results to the client until all Regions are scattered. If its value is set to 1 (by default), TiDB returns the results only after the scattering is completed. If its value is set to 0, TiDB returns the results regardless of the scattering status.
- `tidb_wait_split_region_timeout`: This variable is to set the execution timeout of the `SPLIT REGION` statement, in seconds. The default value is 300s. If the `split` operation is not completed within the duration, TiDB returns a timeout error.

Split Table Region

The key of row data in each table is encoded by `table_id` and `row_id`. The format is as follows:

```
t[table_id]_r[row_id]
```

For example, when `table_id` is 22 and `row_id` is 11:

```
t22_r11
```

Row data in the same table have the same `table_id`, but each row has its unique `row_id` that can be used for Region split.

Even Split

Because `row_id` is an integer, the value of the key to be split can be calculated according to the specified `lower_value`, `upper_value`, and `region_num`. TiDB first calculates the step value ($\text{step} = (\text{upper_value} - \text{lower_value})/\text{region_num}$). Then split will be done evenly per each “step” between `lower_value` and `upper_value` to generate the number of Regions as specified by `region_num`.

For example, if you want 16 evenly split Regions split from key `rangeminInt64~maxInt64` for table t, you can use this statement:

```
SPLIT TABLE t BETWEEN (-9223372036854775808) AND (9223372036854775807)
→ REGIONS 16;
```

This statement splits table t into 16 Regions between `minInt64` and `maxInt64`. If the given primary key range is smaller than the specified one, for example, `0~1000000000`, you can use `0` and `1000000000` take place of `minInt64` and `maxInt64` respectively to split Regions.

```
SPLIT TABLE t BETWEEN (0) AND (1000000000) REGIONS 16;
```

Uneven split

If the known data is unevenly distributed, and you want a Region to be split respectively in key ranges `-inf ~ 10000`, `10000 ~ 90000`, and `90000 ~ +inf`, you can achieve this by setting fixed points, as shown below:

```
SPLIT TABLE t BY (10000), (90000);
```

Split index Region

The key of the index data in the table is encoded by `table_id`, `index_id`, and the value of the index column. The format is as follows:

```
t[table_id]_i[index_id][index_value]
```

For example, when `table_id` is 22, `index_id` is 5, and `index_value` is abc:

```
t22_i5abc
```

The `table_id` and `index_id` of the same index data in one table is the same. To split index Regions, you need to split Regions based on `index_value`.

Even Spilt

The way to split index evenly works the same as splitting data evenly. However, calculating the value of step is more complicated, because `index_value` might not be an integer.

The values of `upper` and `lower` are encoded into a byte array firstly. After removing the longest common prefix of `lower` and `upper` byte array, the first 8 bytes of `lower` and `upper` are converted into the `uint64` format. Then $\text{step} = (\text{upper} - \text{lower})/\text{num}$ is calculated. After that, the calculated step is encoded into a byte array, which is appended to the longest common prefix of the `lower` and `upper` byte array for index split. Here is an example:

If the column of the `idx` index is of the integer type, you can use the following SQL statement to split index data:

```
SPLIT TABLE t INDEX idx BETWEEN (-9223372036854775808) AND
    ↪ (9223372036854775807) REGIONS 16;
```

This statement splits the Region of index `idx` in table `t` into 16 Regions from `minInt64` to `maxInt64`.

If the column of index `idx1` is of varchar type, and you want to split index data by prefix letters.

```
SPLIT TABLE t INDEX idx1 BETWEEN ("a") AND ("z") REGIONS 25;
```

This statement splits index `idx1` into 25 Regions from `a~z`. The range of Region 1 is `[minIndexValue, b)`; the range of Region 2 is `[b, c)`; ... the range of Region 25 is `[y, ↪ minIndexValue]`. For the `idx` index, data with the `a` prefix is written into Region 1, and data with the `b` prefix is written into Region 2, and so on.

In the split method above, both data with the `y` and `z` prefixes are written into Region 25, because the upper bound is not `z`, but `{` (the character next to `z` in ASCII). Therefore, a more accurate split method is as follows:

```
SPLIT TABLE t INDEX idx1 BETWEEN ("a") AND ("{") REGIONS 26;
```

This statement splits index `idx1` of the table `t` into 26 Regions from `a~{`. The range of Region 1 is `[minIndexValue, b)`; the range of Region 2 is `[b, c)`; ... the range of Region 25 is `[y, z)`, and the range of Region 26 is `[z, maxIndexValue)`.

If the column of index `idx2` is of time type like timestamp/datetime, and you want to split index Region by year:

```
SPLIT TABLE t INDEX idx2 BETWEEN ("2010-01-01 00:00:00") AND ("2020-01-01
    ↪ 00:00:00") REGIONS 10;
```

This statement splits the Region of index `idx2` in table `t` into 10 Regions from `2010-01-01 00:00:00` to `2020-01-01 00:00:00`. The range of Region 1 is `[minIndexValue ↪ , 2011-01-01 00:00:00)`; the range of Region 2 is `[2011-01-01 00:00:00, ↪ 2012-01-01 00:00:00)` and so on.

If you want to split the index Region by day, see the following example:

```
SPLIT TABLE t INDEX idx2 BETWEEN ("2020-06-01 00:00:00") AND ("2020-07-01
    ↪ 00:00:00") REGIONS 30;
```

This statement splits the data of June 2020 of index `idx2` in table `t` into 30 Regions, each Region representing 1 day.

Region split methods for other types of index columns are similar.

For data Region split of joint indexes, the only difference is that you can specify multiple columns values.

For example, index `idx3 (a, b)` contains 2 columns, with column `a` of timestamp type and column `b` int. If you just want to do a time range split according to column `a`, you can use the SQL statement for splitting time index of a single column. In this case, do not specify the value of column `b` in `lower_value` and `upper_value`.

```
SPLIT TABLE t INDEX idx3 BETWEEN ("2010-01-01 00:00:00") AND ("2020-01-01
↪ 00:00:00") REGIONS 10;
```

Within the same range of time, if you want to do one more split according to column `b` column. Just specify the value for column `b` when splitting.

```
SPLIT TABLE t INDEX idx3 BETWEEN ("2010-01-01 00:00:00", "a") AND ("2010-01-01 00:00:00", "z") REGIONS 10;
```

This statement splits 10 Regions in the range of `a~z` according to the value of column `b`, with the same time prefix as column `a`. If the value specified for column `a` is different, the value of column `b` might not be used in this case.

Uneven Split

Index data can also be split by specified index values.

For example, there is `idx4 (a,b)`, with column `a` of the varchar type and column `b` of the timestamp type.

```
SPLIT TABLE t1 INDEX idx4 BY ("a", "2000-01-01 00:00:01"), ("b", "
↪ 2019-04-17 14:26:19"), ("c", "");
```

This statement specifies 3 values to split 4 Regions. The range of each Region is as follows:

<code>region1 [minIndexValue</code>	<code>, ("a", "2000-01-01 00:00:01"))</code>
<code>region2 [("a", "2000-01-01 00:00:01") , ("b", "2019-04-17 14:26:19"))</code>	
<code>region3 [("b", "2019-04-17 14:26:19") , ("c", ""))</code>	
<code>region4 [("c", "") , maxIndexValue)</code>	

Split Regions for partitioned tables

Splitting Regions for partitioned tables is the same as splitting Regions for ordinary tables. The only difference is that the same split operation is performed for every partition.

- The syntax of even split:

```
SPLIT [PARTITION] TABLE t [PARTITION] [(partition_name_list...)] [INDEX
↪ index_name] BETWEEN (lower_value) AND (upper_value) REGIONS
↪ region_num
```

- The syntax of uneven split:

```
SPLIT [PARTITION] TABLE table_name [PARTITION (partition_name_list...)]
→ [INDEX index_name] BY (value_list) [, (value_list)] ...
```

Examples of Split Regions for partitioned tables

- Create a partitioned table `t`. Suppose that you want to create a Hash table divided into two partitions. The example statement is as follows:

```
create table t (a int,b int,index idx(a)) partition by hash(a)
→ partitions 2;
```

After creating the table `t`, a Region is split for each partition. Use the `SHOW TABLE REGIONS` syntax to view the Regions of this table:

```
show table t regions;
```

REGION_ID	START_KEY	END_KEY	LEADER_ID	LEADER_STORE_ID	PEERS	SCATTERING	WRITTEN_BYTES	READ_BYTES	APPROXIMATE_SIZE(MB)	APPROXIMATE_KEYS
1978	t_1400_	t_1401_	1979	4	1979, 1980,					
1981	0	0	0	1	10					

REGION_ID	START_KEY	END_KEY	LEADER_ID	LEADER_STORE_ID	PEERS	SCATTERING	WRITTEN_BYTES	READ_BYTES	APPROXIMATE_SIZE(MB)	APPROXIMATE_KEYS
6	t_1401_		17	4	17, 18, 21					
0	223	0	0	1	10					

- Use the `SPLIT` syntax to split a Region for each partition. Suppose that you want to split the data in the $[0, 10000]$ range of each partition into four Regions. The example statement is as follows:

```
split partition table t between (0) and (10000) regions 4;
```

In the above statement, 0 and 10000 respectively represent the `row_id` of the upper and lower boundaries corresponding to the hotspot data you want to scatter.

Note:

This example only applies to scenarios where hotspot data is evenly distributed. If the hotspot data is unevenly distributed in a specified data range, refer to the syntax of uneven split in [Split Regions for partitioned tables](#).

3. Use the `SHOW TABLE REGIONS` syntax to view the Regions of this table again. You can see that this table now has ten Regions, each partition with five Regions, four of which are the row data and one is the index data.

```
show table t regions;
```

REGION_ID	START_KEY	END_KEY	LEADER_ID	LEADER_STORE_ID	PEERS	SCATTERING	WRITTEN_BYTES	READ_BYTES	APPROXIMATE_SIZE(MB)	APPROXIMATE_KEYS
1998	t_1400_r	t_1400_r_2500	2001	5	2000, 2001, 2015	0	132	0	1	0
2006	t_1400_r_2500	t_1400_r_5000	2016	1	2007, 2016, 2017	0	35	0	1	0
2010	t_1400_r_5000	t_1400_r_7500	2012	2	2011, 2012, 2013	0	35	0	1	0
1978	t_1400_r_7500	t_1401_	1979	4	1979, 1980, 1981	0	621	0	1	0
1982	t_1400_	t_1400_r	2014	3	1983, 1984, 2014	0	35	0	1	0
1990	t_1401_r	t_1401_r_2500	1992	2	1991, 1992, 2020	0	120	0	1	0
1994	t_1401_r_2500	t_1401_r_5000	1997	5	1996, 1997, 2021	0	129	0	1	0
2002	t_1401_r_5000	t_1401_r_7500	2003	4	2003, 2023, 2022	0	141	0	1	0

↓		0					
6	t_1401_r_7500		17	4		17,	
↓	0	601	0	1			
↓	0						
1986	t_1401_	t_1401_r	1989	5			
↓	0	123	0	1			
↓	0						
+--							
↓	-----+-----+-----+-----+						
↓							

4. You can also split Regions for the index of each partition. For example, you can split the [1000,10000] range of the idx index into two Regions. The example statement is as follows:

```
split partition table t index idx between (1000) and (10000) regions
  ↓
  2;
```

Examples of Split Region for a single partition

You can specify the partition to be split.

1. Create a partitioned table. Suppose that you want to create a Range partitioned table split into three partitions. The example statement is as follows:

```
create table t ( a int, b int, index idx(b)) partition by range( a ) (
  partition p1 values less than (10000),
  partition p2 values less than (20000),
  partition p3 values less than (MAXVALUE) );
```

2. Suppose that you want to split the data in the [0,10000] range of the p1 partition into two Regions. The example statement is as follows:

```
split partition table t partition (p1) between (0) and (10000) regions
  ↓
  2;
```

3. Suppose that you want to split the data in the [10000,20000] range of the p2 partition into two Regions. The example statement is as follows:

```
split partition table t partition (p2) between (10000) and (20000)
  ↓
  regions 2;
```

4. You can use the SHOW TABLE REGIONS syntax to view the Regions of this table:

```
show table t regions;
```

REGION_ID	START_KEY	END_KEY	LEADER_ID	LEADER_STORE_ID	PEERS	SCATTERING	WRITTEN_BYTES	READ_BYTES	APPROXIMATE_SIZE(MB)	APPROXIMATE_KEYS
2040	t_1406_	t_1406_r_5000	2045	3	2043, 2045, 2044	0	0	0	1	0
2032	t_1406_r_5000	t_1407_	2033	4	2033, 2034, 2035	0	0	0	1	0
2046	t_1407_	t_1407_r_15000	2048	2	2047, 2048, 2050	0	35	0	1	0
2036	t_1407_r_15000	t_1408_	2037	4	2037, 2038, 2039	0	0	0	1	0
6	t_1408_		17	4	17, 18, 21	0	214	0	1	0

5. Suppose that you want to split the [0,20000] range of the `idx` index of the p1 and p2 partitions into two Regions. The example statement is as follows:

```
split partition table t partition (p1,p2) index idx between (0) and
    ↵ (20000) regions 2;
```

11.5.2.120.3 pre_split_regions

To have evenly split Regions when a table is created, it is recommended you use `SHARD_ROW_ID_BITS` together with `PRE_SPLIT_REGIONS`. When a table is created successfully, `PRE_SPLIT_REGIONS` pre-splits tables into the number of Regions as specified by $2^{\lceil \text{PRE_SPLIT_REGIONS} \rceil}$.

Note:

The value of PRE_SPLIT_REGIONS must be less than or equal to that of SHARD_ROW_ID_BITS.

The `tidb_scatter_region` global variable affects the behavior of PRE_SPLIT_REGIONS → . This variable controls whether to wait for Regions to be pre-split and scattered before returning results after the table creation. If there are intensive writes after creating the table, you need to set the value of this variable to 1, then TiDB will not return the results to the client until all the Regions are split and scattered. Otherwise, TiDB writes the data before the scattering is completed, which will have a significant impact on write performance.

Examples of pre_split_regions

```
create table t (a int, b int, index idx1(a)) shard_row_id_bits = 4
    → pre_split_regions=2;
```

After building the table, this statement splits $4 + 1$ Regions for table t. 4 (2^2) Regions are used to save table row data, and 1 Region is for saving the index data of `idx1`.

The ranges of the 4 table Regions are as follows:

```
region1: [ -inf      , 1<<61 )
region2: [ 1<<61    , 2<<61 )
region3: [ 2<<61    , 3<<61 )
region4: [ 3<<61    , +inf )
```

11.5.2.120.4 Notes

The Region split by the Split Region statement is controlled by the `Region merge` scheduler in PD. To avoid PD re-merging the newly split Region soon after, you need to `dynamically modify` configuration items related to the Region merge feature.

11.5.2.120.5 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.120.6 See also

- `SHOW TABLE REGIONS`
- Session variables: `tidb_scatter_region`, `tidb_wait_split_region_finish` and `tidb_wait_split_region_timeout`.

11.5.2.121 START TRANSACTION

This statement starts a new transaction inside of TiDB. It is similar to the statement BEGIN.

In the absence of a START TRANSACTION statement, every statement will by default autocommit in its own transaction. This behavior ensures MySQL compatibility.

11.5.2.121.1 Synopsis

BeginTransactionStmt:

```
BeginTransactionStmt ::=  
    'BEGIN' ( 'PESSIMISTIC' | 'OPTIMISTIC' )?  
  | 'START' 'TRANSACTION' ( 'READ' ( 'WRITE' | 'ONLY' ( ( 'WITH' 'TIMESTAMP'  
    ↪ 'BOUND' TimestampBound )? | AsOfClause ) ) | 'WITH' 'CONSISTENT'  
    ↪ 'SNAPSHOT' | 'WITH' 'CAUSAL' 'CONSISTENCY' 'ONLY' )?  
  
AsOfClause ::=  
    ( 'AS' 'OF' 'TIMESTAMP' Expression)
```

11.5.2.121.2 Examples

```
mysql> CREATE TABLE t1 (a int NOT NULL PRIMARY KEY);  
Query OK, 0 rows affected (0.12 sec)  
  
mysql> START TRANSACTION;  
Query OK, 0 rows affected (0.00 sec)  
  
mysql> INSERT INTO t1 VALUES (1);  
Query OK, 1 row affected (0.00 sec)  
  
mysql> COMMIT;  
Query OK, 0 rows affected (0.01 sec)
```

11.5.2.121.3 MySQL compatibility

- START TRANSACTION immediately starts a transaction inside TiDB. This differs from MySQL, where START TRANSACTION lazily creates a transaction. But START TRANSACTION in TiDB is equivalent to MySQL's START TRANSACTION WITH CONSISTENT SNAPSHOT.
- The statement START TRANSACTION READ ONLY is parsed for compatibility with MySQL, but still allows write operations.

11.5.2.121.4 See also

- COMMIT
- ROLLBACK
- BEGIN
- START TRANSACTION WITH CAUSAL CONSISTENCY ONLY

11.5.2.122 TABLE

The TABLE statement can be used instead of SELECT * FROM when no aggregation or complex filtering is needed.

11.5.2.122.1 Synopsis

```
TableStmt ::=  
    "TABLE" Table ( "ORDER BY" Column )? ( "LIMIT" NUM )?
```

11.5.2.122.2 Examples

```
CREATE TABLE t1(id INT PRIMARY KEY);
```

```
Query OK, 0 rows affected (0.31 sec)
```

```
INSERT INTO t1 VALUES (1),(2),(3);
```

```
Query OK, 3 rows affected (0.06 sec)  
Records: 3 Duplicates: 0 Warnings: 0
```

```
TABLE t1;
```

```
+----+  
| id |  
+----+  
| 1 |  
| 2 |  
| 3 |  
+----+  
3 rows in set (0.01 sec)
```

```
TABLE t1 ORDER BY id DESC;
```

```
+----+
| id |
+----+
| 3 |
| 2 |
| 1 |
+----+
3 rows in set (0.01 sec)
```

```
TABLE t1 LIMIT 1;
```

```
+----+
| id |
+----+
| 1 |
+----+
1 row in set (0.01 sec)
```

11.5.2.122.3 MySQL compatibility

The TABLE statement was introduced in MySQL 8.0.19.

11.5.2.122.4 See also

- [SELECT](#)
- [TABLE statements in MySQL](#)

11.5.2.123 TRACE

The TRACE statement provides detailed information about query execution. It is intended to be viewed through a Graphical interface exposed by the TiDB server's status port.

11.5.2.123.1 Synopsis

TraceStmt:

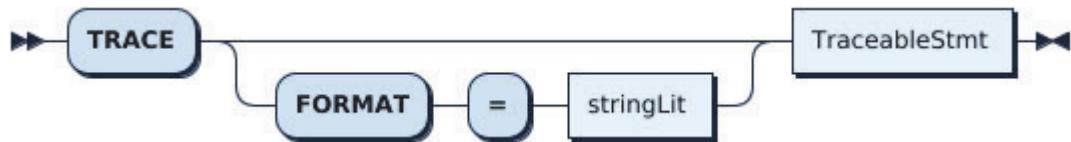


Figure 319: TraceStmt

TraceableStmt:

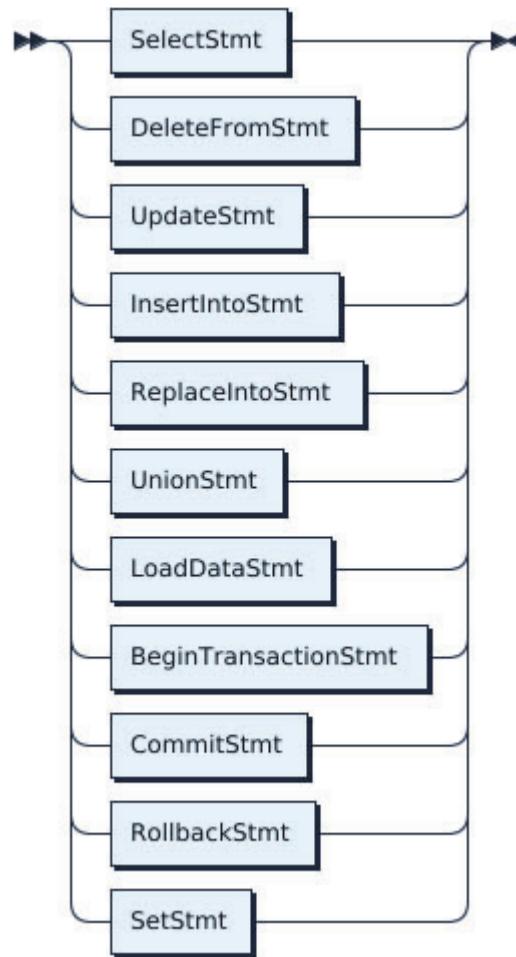


Figure 320: TraceableStmt

11.5.2.123.2 Examples

```
trace format='row' select * from mysql.user;
```

operation	startTS	duration
trace	17:03:31.938237	886.086µs
session.Execute	17:03:31.938247	507.812µs
session.ParseSQL	17:03:31.938254	22.504µs
executor.Compile	17:03:31.938321	278.931µs
session.getTxnFuture	17:03:31.938337	1.515µs
session.runStmt	17:03:31.938613	109.578µs
TableReaderExecutor.Open	17:03:31.938645	50.657µs
distsql.Select	17:03:31.938666	21.066µs

```

|      - RPCClient.SendRequest      | 17:03:31.938799 | 158.411µs |
|      - session.CommitTxn        | 17:03:31.938705 | 12.06µs |
|      - session.doCommitWithRetry | 17:03:31.938709 | 2.437µs |
| -* executor.TableReaderExecutor.Next | 17:03:31.938781 | 224.327µs |
| -* executor.TableReaderExecutor.Next | 17:03:31.939019 | 6.266µs |
+-----+-----+-----+
    ↵
13 rows in set (0.00 sec)

```

```
trace format='json' select * from mysql.user;
```

The JSON formatted trace can be pasted into the trace viewer, which is accessed via the TiDB status port:

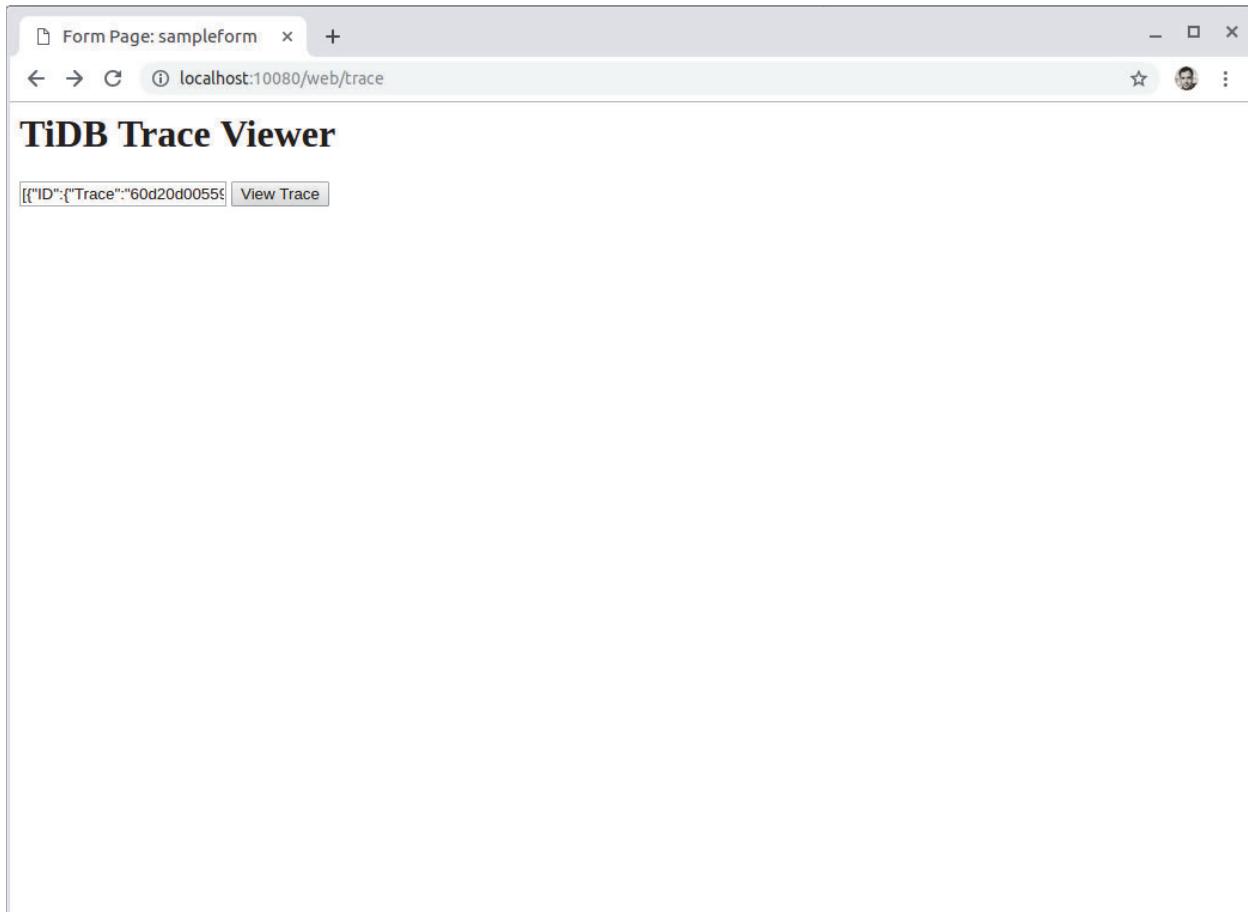


Figure 321: TiDB Trace Viewer-1

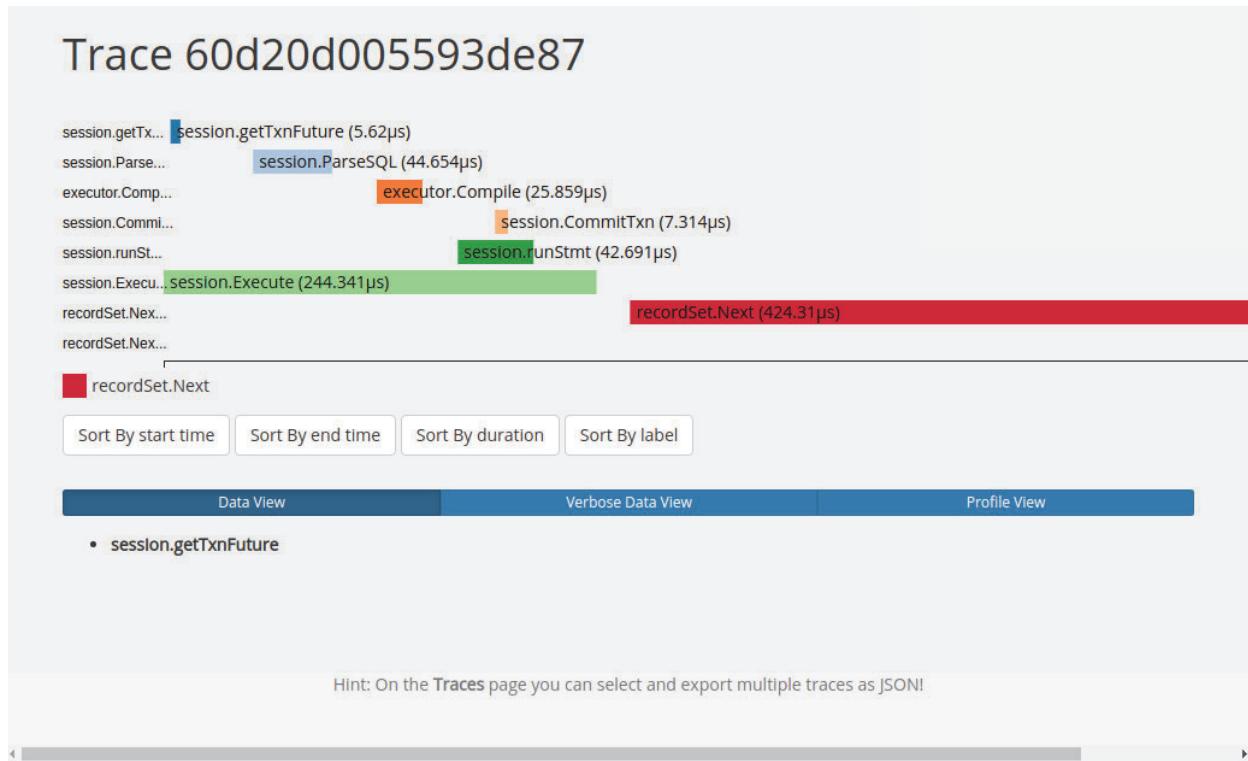


Figure 322: TiDB Trace Viewer-2

11.5.2.123.3 MySQL compatibility

This statement is a TiDB extension to MySQL syntax.

11.5.2.123.4 See also

- EXPLAIN ANALYZE

11.5.2.124 TRUNCATE

The TRUNCATE statement removes all data from the table in a non-transactional way. TRUNCATE can be thought of as semantically the same as `DROP TABLE + CREATE TABLE` with the previous definition.

Both `TRUNCATE TABLE tableName` and `TRUNCATE tableName` are valid syntax.

11.5.2.124.1 Synopsis

`TruncateTableStmt:`

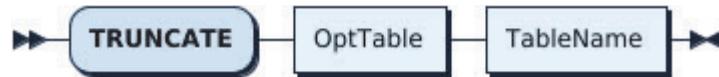


Figure 323: TruncateTableStmt

OptTable:



Figure 324: OptTable

TableName:

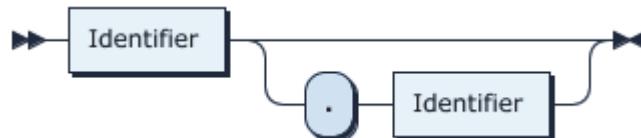


Figure 325: TableName

11.5.2.124.2 Examples

```

mysql> CREATE TABLE t1 (a INT NOT NULL PRIMARY KEY);
Query OK, 0 rows affected (0.11 sec)

mysql> INSERT INTO t1 VALUES (1),(2),(3),(4),(5);
Query OK, 5 rows affected (0.01 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM t1;
+---+
| a |
+---+
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
+---+
5 rows in set (0.00 sec)

mysql> TRUNCATE t1;
Query OK, 0 rows affected (0.11 sec)

```

```

mysql> SELECT * FROM t1;
Empty set (0.00 sec)

mysql> INSERT INTO t1 VALUES (1),(2),(3),(4),(5);
Query OK, 5 rows affected (0.01 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> TRUNCATE TABLE t1;
Query OK, 0 rows affected (0.11 sec)

```

11.5.2.124.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.124.4 See also

- [DROP TABLE](#)
- [DELETE](#)
- [CREATE TABLE](#)
- [SHOW CREATE TABLE](#)

11.5.2.125 UPDATE

The UPDATE statement is used to modify data in a specified table.

11.5.2.125.1 Synopsis

UpdateStmt:

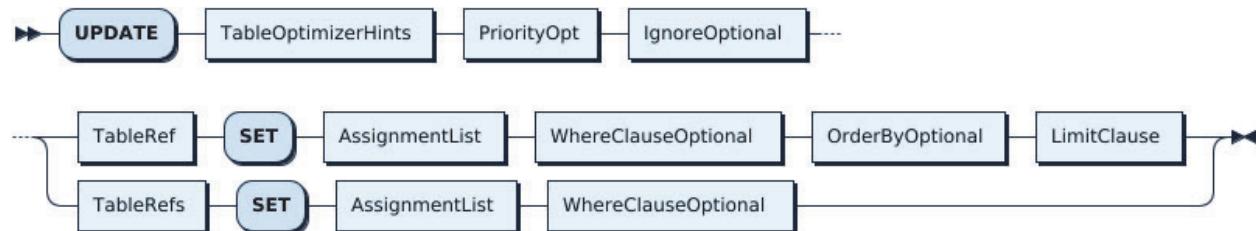


Figure 326: UpdateStmt

PriorityOpt:



Figure 327: PriorityOpt

TableRef:



Figure 328: TableRef

TableRefs:



Figure 329: TableRefs

AssignmentList:

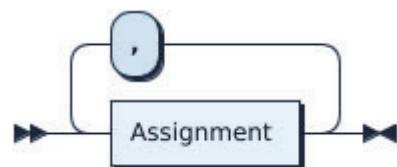


Figure 330: AssignmentList

WhereClauseOptional:



Figure 331: WhereClauseOptional

11.5.2.125.2 Examples

```

mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY AUTO_INCREMENT, c1 INT
   ↪ NOT NULL);
Query OK, 0 rows affected (0.11 sec)

mysql> INSERT INTO t1 (c1) VALUES (1), (2), (3);
Query OK, 3 rows affected (0.02 sec)
Records: 3 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM t1;
+----+---+
| id | c1 |
+----+---+
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
+----+---+
3 rows in set (0.00 sec)

mysql> UPDATE t1 SET c1=5 WHERE c1=3;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1 Changed: 1 Warnings: 0

mysql> SELECT * FROM t1;
+----+---+
| id | c1 |
+----+---+
| 1 | 1 |
| 2 | 2 |
| 3 | 5 |
+----+---+
3 rows in set (0.00 sec)

```

11.5.2.125.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.125.4 See also

- [INSERT](#)
- [SELECT](#)
- [DELETE](#)
- [REPLACE](#)

11.5.2.126 USE

The USE statement selects a current database for the user session.

11.5.2.126.1 Synopsis

UseStmt:

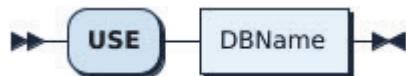


Figure 332: UseStmt

DBName:



Figure 333: DBName

11.5.2.126.2 Examples

```

mysql> USE mysql;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> SHOW TABLES;
+-----+
| Tables_in_mysql      |
+-----+
| GLOBAL_VARIABLES    |
| bind_info            |
| columns_priv         |
| db                   |
| default_roles        |
| expr_pushdown_blacklist |
| gc_delete_range      |
| gc_delete_range_done |
| global_priv          |
| help_topic           |
| opt_rule_blacklist   |
| role_edges           |
| stats_buckets         |
| stats_feedback        |
| stats_histograms      |
+-----+
  
```

```

| stats_meta      |
| stats_top_n    |
| tables_priv    |
| tidb           |
| user           |
+-----+
20 rows in set (0.01 sec)

mysql> CREATE DATABASE newtest;
Query OK, 0 rows affected (0.10 sec)

mysql> USE newtest;
Database changed
mysql> SHOW TABLES;
Empty set (0.00 sec)

mysql> CREATE TABLE t1 (a int);
Query OK, 0 rows affected (0.10 sec)

mysql> SHOW TABLES;
+-----+
| Tables_in_newtest |
+-----+
| t1                |
+-----+
1 row in set (0.00 sec)

```

11.5.2.126.3 MySQL compatibility

This statement is understood to be fully compatible with MySQL. Any compatibility differences should be reported via an issue on GitHub.

11.5.2.126.4 See also

- [CREATE DATABASE](#)
- [SHOW TABLES](#)

11.5.2.127 WITH

A Common Table Expression (CTE) is a temporary result set that can be referred multiple times within a SQL statement to improve the statement's readability and execution efficiency. You can apply the WITH statement to use Common Table Expressions.

11.5.2.127.1 Synopsis

WithClause:

```
WithClause ::=  
    "WITH" WithList  
  |  "WITH" recursive WithList
```

WithList:

```
WithList ::=  
    WithList ',' CommonTableExpr  
  |  CommonTableExpr
```

CommonTableExpr:

```
CommonTableExpr ::=  
    Identifier IdentListWithParenOpt "AS" SubSelect
```

IdentListWithParenOpt:

```
IdentListWithParenOpt ::=  
  |  '(' IdentList ')' 
```

11.5.2.127.2 Examples

Non-recursive CTE:

```
WITH CTE AS (SELECT 1, 2) SELECT * FROM cte t1, cte t2;
```

```
+---+---+---+---+  
| 1 | 2 | 1 | 2 |  
+---+---+---+---+  
| 1 | 2 | 1 | 2 |  
+---+---+---+---+  
1 row in set (0.00 sec)
```

Recursive CTE:

```
WITH RECURSIVE cte(a) AS (SELECT 1 UNION SELECT a+1 FROM cte WHERE a < 5)  
  ↪ SELECT * FROM cte;
```

```
+---+  
| a |  
+---+  
| 1 |  
| 2 |  
| 3 |
```

```
| 4 |
| 5 |
+---+
5 rows in set (0.00 sec)
```

11.5.2.127.3 MySQL compatibility

- In strict mode, when the data length recursively calculated exceeds the data length of the seed part, TiDB returns a warning while MySQL returns an error. In non-strict mode, the behavior of TiDB is consistent with that of MySQL.
- The data type for recursive CTE is determined by the seed part. The data type of the seed part is not completely consistent with MySQL in some cases (such as functions).
- In the case of multiple UNION / UNION ALL operators, MySQL does not allow UNION to be followed by UNION ALL, but TiDB does.
- If there is a problem with the definition of a CTE, TiDB will report an error, while MySQL will not if the CTE is not referred.

11.5.2.127.4 See also

- [SELECT](#)
- [INSERT](#)
- [DELETE](#)
- [UPDATE](#)
- [REPLACE](#)

11.5.3 Data Types

11.5.3.1 Data Types

TiDB supports all the data types in MySQL except the **SPATIAL** type. This includes all the [numeric types](#), [string types](#), [date & time types](#), and [the JSON type](#).

The definitions used for datatypes are specified as $T(M[, D])$. Where by:

- T indicates the specific data type.
- M indicates the maximum display width for integer types. For floating-point and fixed-point types, M is the total number of digits that can be stored (the precision). For string types, M is the maximum length. The maximum permissible value of M depends on the data type.
- D applies to floating-point and fixed-point types and indicates the number of digits following the decimal point (the scale).
- **fsp** applies to the **TIME**, **DATETIME**, and **TIMESTAMP** types and represents the fractional seconds precision. The **fsp** value, if given, must be in the range 0 to 6. A value of 0 signifies that there is no fractional part. If omitted, the default precision is 0.

11.5.3.2 Default Values

The `DEFAULT` value clause in a data type specification indicates a default value for a column. The default value must be a constant and cannot be a function or an expression. But for the time type, you can specify the `NOW`, `CURRENT_TIMESTAMP`, `LOCALTIME`, and `LOCALTIMESTAMP` functions as the default for `TIMESTAMP` and `DATETIME` columns.

The `BLOB`, `TEXT`, and `JSON` columns **cannot** be assigned a default value.

If a column definition includes no explicit `DEFAULT` value, TiDB determines the default value as follows:

- If the column can take `NULL` as a value, the column is defined with an explicit `DEFAULT ↵ NULL` clause.
- If the column cannot take `NULL` as the value, TiDB defines the column with no explicit `DEFAULT` clause.

For data entry into a `NOT NULL` column that has no explicit `DEFAULT` clause, if an `INSERT` or `REPLACE` statement includes no value for the column, TiDB handles the column according to the SQL mode in effect at the time:

- If strict SQL mode is enabled, an error occurs for transactional tables, and the statement is rolled back. For nontransactional tables, an error occurs.
- If strict mode is not enabled, TiDB sets the column to the implicit default value for the column data type.

Implicit defaults are defined as follows:

- For numeric types, the default is 0. If declared with the `AUTO_INCREMENT` attribute, the default is the next value in the sequence.
- For date and time types other than `TIMESTAMP`, the default is the appropriate “zero” value for the type. For `TIMESTAMP`, the default value is the current date and time.
- For string types other than `ENUM`, the default value is the empty string. For `ENUM`, the default is the first enumeration value.

11.5.3.3 Numeric Types

TiDB supports all the MySQL numeric types, including:

- [Integer Types](#) (Exact Value)
- [Floating-Point Types](#) (Approximate Value)
- [Fixed-Point Types](#) (Exact Value)

11.5.3.3.1 Integer types

TiDB supports all the MySQL integer types, including INTEGER/INT, TINYINT, SMALLINT, MEDIUMINT, and BIGINT. For more information, see [Integer Data Type Syntax in MySQL](#).

The following table summarizes field descriptions:

Syntax Element	Description
M	the display width of the type. Optional.
UNSIGNED	UNSIGNED. If omitted, it is SIGNED.
ZEROFILL	If you specify ZEROFILL for a numeric column, TiDB automatically adds the UNSIGNED attribute to the column.

The following table summarizes the required storage and range for integer types supported by TiDB:

Data Type	Storage Required (bytes)	Minimum Value (signed/unsigned)	Maximum Value (signed/unsigned)
TINYINT	1	-128 / 0	127 / 255
SMALLINT	2	-32768 / 0	32767 / 65535
MEDIUMINT	3	-8388608 / 0	8388607 / 16777215
INT	4	-2147483648 / 0	2147483647 / 4294967295
BIGINT	8	-9223372036854775808 / 0	9223372036854775807 / 18446744073709551615

BIT type

The BIT data type. A type of BIT(M) enables the storage of M-bit values. M can range from 1 to 64, with the default value of 1:

BIT[(M)]

BOOLEAN type

The BOOLEAN type and its alias BOOL are equivalent to TINYINT(1). If the value is 0, it is considered as `False`; otherwise, it is considered `True`. As in MySQL, `True` is 1 and `False` is 0:

BOOLEAN

TINYINT type

The TINYINT data type stores signed values of range [-128, 127] and unsigned values of range [0, 255]:

```
TINYINT[(M)] [UNSIGNED] [ZEROFILL]
```

SMALLINT type

The SMALLINT data type stores signed values of range [-32768, 32767], and unsigned values of range [0, 65535]:

```
SMALLINT[(M)] [UNSIGNED] [ZEROFILL]
```

MEDIUMINT type

The MEDIUMINT data type stores signed values of range [-8388608, 8388607], and unsigned values of range [0, 16777215]:

```
MEDIUMINT[(M)] [UNSIGNED] [ZEROFILL]
```

INTEGER type

The INTEGER type and its alias INT stores signed values of range [-2147483648, 2147483647], and unsigned values of range [0, 4294967295]:

```
INT[(M)] [UNSIGNED] [ZEROFILL]
```

You can also use another form:

```
INTEGER[(M)] [UNSIGNED] [ZEROFILL]
```

BIGINT type

The BIGINT data type stores signed values of range [-9223372036854775808, 9223372036854775807], and unsigned values of range [0, 18446744073709551615]:

```
BIGINT[(M)] [UNSIGNED] [ZEROFILL]
```

11.5.3.3.2 Floating-point types

TiDB supports all the MySQL floating-point types, including FLOAT, and DOUBLE. For more information, see [Floating-Point Types \(Approximate Value\) - FLOAT, DOUBLE in MySQL](#).

The following table summarizes field descriptions:

Syntax Element	Description
M	the total number of digits
D	the number of digits following the decimal point
UNSIGNED	UNSIGNED. If omitted, it is SIGNED.
ZEROFILL	If you specify ZEROFILL for a numeric column, TiDB automatically adds the UNSIGNED attribute to the column.

The following table summarizes the required storage for floating-point types supported by TiDB:

Data Type	Storage Required (bytes)
FLOAT	4
FLOAT(p)	If $0 \leq p \leq 24$, it is 4; if $25 \leq p \leq 53$, it is 8
DOUBLE	8

FLOAT type

The FLOAT type stores a single-precision floating-point number. Permissible values are -3.402823466E+38 to -1.175494351E-38, 0, and 1.175494351E-38 to 3.402823466E+38. These are the theoretical limits, based on the IEEE standard. The actual range might be slightly smaller depending on your hardware or operating system.

FLOAT(p) can be used to represent the required precision in bits. TiDB uses this value only to determine whether to use FLOAT or DOUBLE for the resulting data type. If p is from 0 to 24, the data type becomes FLOAT with no M or D values. If p is from 25 to 53, the data type becomes DOUBLE with no M or D values. The range of the resulting column is the same as for the single-precision FLOAT or double-precision DOUBLE data type.

```
FLOAT[(M,D)] [UNSIGNED] [ZEROFILL]
FLOAT(p) [UNSIGNED] [ZEROFILL]
```

Note:

As in MySQL, the FLOAT data type stores approximate values. For values such as currency, it is recommended to use the DECIMAL type instead. In TiDB, the default precision of the FLOAT data type is 8 bits, but in MySQL, the

default precision is 6 bits. For example, assuming that you insert 123456789 and 1.23456789 into columns of the FLOAT type in both TiDB and MySQL, when you query the corresponding values in MySQL, you get 123457000 and 1.23457, while in TiDB, you get 123456790 and 1.2345679.

DOUBLE type

The DOUBLE type, and its alias DOUBLE PRECISION stores a double-precision floating-point number. Permissible values are -1.7976931348623157E+308 to -2.2250738585072014E-308, 0, and 2.2250738585072014E-308 to 1.7976931348623157E+308. These are the theoretical limits, based on the IEEE standard. The actual range might be slightly smaller depending on your hardware or operating system.

```
DOUBLE[(M,D)] [UNSIGNED] [ZEROFILL]
DOUBLE PRECISION [(M,D)] [UNSIGNED] [ZEROFILL], REAL[(M,D)] [UNSIGNED] [
    ↪ ZEROFILL]
```

Warning:

As in MySQL, the DOUBLE data type stores approximate values. For values such as currency, it is recommended to use the DECIMAL type instead.

11.5.3.3.3 Fixed-point types

TiDB supports all the MySQL floating-point types, including DECIMAL, and NUMERIC. For more information, [Fixed-Point Types \(Exact Value\) - DECIMAL, NUMERIC in MySQL](#).

The meaning of the fields:

Syntax		
Element	Description	
M	the total number of digits	
D	the number of digits after the decimal point	
UNSIGNED	UNSIGNED. If omitted, it is SIGNED.	
ZEROFILL	If you specify ZEROFILL for a numeric column, TiDB automatically adds the UNSIGNED attribute to the column.	

Element	Syntax	Description
---------	--------	-------------

DECIMAL type

DECIMAL and its alias NUMERIC stores a packed “exact” fixed-point number. M is the total number of digits (the precision), and D is the number of digits after the decimal point (the scale). The decimal point and (for negative numbers) the - sign are not counted in M. If D is 0, values have no decimal point or fractional part. The maximum number of digits (M) for DECIMAL is 65. The maximum number of supported decimals (D) is 30. If D is omitted, the default is 0. If M is omitted, the default is 10.

DECIMAL[(M[,D])] [UNSIGNED] [ZEROFILL]
NUMERIC[(M[,D])] [UNSIGNED] [ZEROFILL]

11.5.3.4 Date and Time Types

TiDB supports all MySQL date and time data types to store temporal values: **DATE**, **TIME**, **DATETIME**, **TIMESTAMP**, and **YEAR**. For more information, see [Date and Time Data Types in MySQL](#).

Each of these types has its range of valid values, and uses a zero value to indicate that it is an invalid value. In addition, the **TIMESTAMP** and **DATETIME** types can automatically generate new time values on modification.

When dealing with date and time value types, note:

- Although TiDB tries to interpret different formats, the date-portion must be in the format of year-month-day (for example, ‘1998-09-04’), rather than month-day-year or day-month-year.
- If the year-portion of a date is specified as 2 digits, TiDB converts it based on [specific rules](#).
- If a numeric value is needed in the context, TiDB automatically converts the date or time value into a numeric type. For example:

```
mysql> SELECT NOW(), NOW()+0, NOW(3)+0;
+-----+-----+-----+
| NOW() | NOW()+0 | NOW(3)+0 |
+-----+-----+-----+
| 2012-08-15 09:28:00 | 20120815092800 | 20120815092800.889 |
+-----+-----+-----+
```

- TiDB might automatically convert invalid values or values beyond the supported range to a zero value of that type. This behavior is dependent on the SQL Mode set. For example:

```

mysql> show create table t1;
+--+
| Table | Create Table
+--+
| t1   | CREATE TABLE `t1` (
  `a` time DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin |
+--+
1 row in set (0.00 sec)

mysql> select @@sql_mode;
+--+
| @@sql_mode
+--+
| ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,
  ↓ ERROR_FOR_DIVISION_BY_ZERO,NO_AUTO_CREATE_USER,
  ↓ NO_ENGINE_SUBSTITUTION |
+--+
1 row in set (0.00 sec)

mysql> insert into t1 values ('2090-11-32:22:33:44');
ERROR 1292 (22007): Truncated incorrect time value:
  ↓ 2090-11-32:22:33:44'
mysql> set @@sql_mode='';
  ↓
  ↓ Query OK, 0 rows affected (0.01 sec)

mysql> insert into t1 values ('2090-11-32:22:33:44');

```

```
Query OK, 1 row affected, 1 warning (0.01 sec)

mysql> select * from t1;
+-----+
| a    |
+-----+
| 00:00:00 |
+-----+
1 row in set (0.01 sec)
```

- Setting different SQL modes can change TiDB behaviors.
- If the SQL mode `NO_ZERO_DATE` is not enabled, TiDB allows month or day in the columns of `DATE` and `DATETIME` to be zero value, for example, ‘2009-00-00’ or ‘2009-01-00’. If this date type is to be calculated in a function, for example, in `DATE_SUB()` or `DATE_ADD()`, the result can be incorrect.
- By default, TiDB enables the SQL mode `NO_ZERO_DATE`. This mode prevents storing zero values such as ‘0000-00-00’.

Different types of zero value are shown in the following table:

Date Type	“Zero” Value
DATE	‘0000-00-00’
TIME	‘00:00:00’
DATETIME	‘0000-00-00 00:00:00’
TIMESTAMP	‘0000-00-00 00:00:00’
YEAR	0000

Invalid `DATE`, `DATETIME`, `TIMESTAMP` values are automatically converted to the corresponding type of zero value (‘0000-00-00’ or ‘0000-00-00 00:00:00’) if the SQL mode permits such usage.

11.5.3.4.1 Supported types

DATE type

`DATE` only contains date-portion and no time-portion, displayed in `YYYY-MM-DD` format. The supported range is ‘1000-01-01’ to ‘9999-12-31’:

DATE

TIME type

For the `TIME` type, the format is `HH:MM:SS[.fraction]` and valid values range from ‘-838:59:59.000000’ to ‘838:59:59.000000’. `TIME` is used not only to indicate the time within

a day but also to indicate the time interval between 2 events. An optional `fsp` value in the range from 0 to 6 may be given to specify fractional seconds precision. If omitted, the default precision is 0:

<code>TIME[(fsp)]</code>

Note:

Pay attention to the abbreviated form of `TIME`. For example, ‘11:12’ means ‘11:12:00’ instead of ‘00:11:12’. However, ‘1112’ means ‘00:11:12’. These differences are caused by the presence or absence of the : character.

DATETIME type

`DATETIME` contains both date-portion and time-portion. Valid values range from ‘1000-01-01 00:00:00.000000’ to ‘9999-12-31 23:59:59.999999’.

TiDB displays `DATETIME` values in `YYYY-MM-DD HH:MM:SS[.fraction]` format, but permits assignment of values to `DATETIME` columns using either strings or numbers. An optional `fsp` value in the range from 0 to 6 may be given to specify fractional seconds precision. If omitted, the default precision is 0:

<code>DATETIME[(fsp)]</code>

TIMESTAMP type

`TIMESTAMP` contains both date-portion and time-portion. Valid values range from ‘1970-01-01 00:00:01.000000’ to ‘2038-01-19 03:14:07.999999’ in UTC time. An optional `fsp` value in the range from 0 to 6 may be given to specify fractional seconds precision. If omitted, the default precision is 0.

In `TIMESTAMP`, zero is not permitted to appear in the month-portion or day-portion. The only exception is zero value itself ‘0000-00-00 00:00:00’.

<code>TIMESTAMP[(fsp)]</code>

Timezone Handling

When `TIMESTAMP` is to be stored, TiDB converts the `TIMESTAMP` value from the current time zone to UTC time zone. When `TIMESTAMP` is to be retrieved, TiDB converts the stored `TIMESTAMP` value from UTC time zone to the current time zone (Note: `DATETIME` is not handled in this way). The default time zone for each connection is the server’s local time zone, which can be modified by the environment variable `time_zone`.

Warning:

As in MySQL, the `TIMESTAMP` data type suffers from the [Year 2038 Problem](#). For storing values that may span beyond 2038, please consider using the `DATETIME` type instead.

YEAR type

The `YEAR` type is specified in the format ‘YYYY’. Supported values range from 1901 to 2155, or the zero value of 0000:

`YEAR[(4)]`

`YEAR` follows the following format rules:

- Four-digit numeral ranges from 1901 to 2155
- Four-digit string ranges from ‘1901’ to ‘2155’
- One-digit or two-digit numeral ranges from 1 to 99. Accordingly, 1-69 is converted to 2001-2069 and 70-99 is converted to 1970-1999
- One-digit or two-digit string ranges from ‘0’ to ‘99’
- Value 0 is taken as 0000 whereas the string ‘0’ or ‘00’ is taken as 2000

Invalid `YEAR` value is automatically converted to 0000 (if users are not using the `NO_ZERO_DATE` SQL mode).

11.5.3.4.2 Automatic initialization and update of `TIMESTAMP` and `DATETIME`

Columns with `TIMESTAMP` or `DATETIME` value type can be automatically initialized or updated to the current time.

For any column with `TIMESTAMP` or `DATETIME` value type in the table, you can set the default or auto-update value as current timestamp.

These properties can be set by setting `DEFAULT CURRENT_TIMESTAMP` and `ON UPDATE CURRENT_TIMESTAMP` when the column is being defined. `DEFAULT` can also be set as a specific value, such as `DEFAULT 0` or `DEFAULT '2000-01-01 00:00:00'`.

```
CREATE TABLE t1 (
    ts TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    dt DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
);
```

The default value for `DATETIME` is `NULL` unless it is specified as `NOT NULL`. For the latter situation, if no default value is set, the default value is be 0.

```
CREATE TABLE t1 (
    dt1 DATETIME ON UPDATE CURRENT_TIMESTAMP, -- default NULL
    dt2 DATETIME NOT NULL ON UPDATE CURRENT_TIMESTAMP -- default 0
);
```

11.5.3.4.3 Decimal part of time value

DATETIME and TIMESTAMP values can contain a fractional part of up to 6 digits which is accurate to milliseconds. In any column of DATETIME or TIMESTAMP types, a fractional part is stored instead of being discarded. With a fractional part, the value is in the format of ‘YYYY-MM-DD HH:MM:SS[.fraction]’, and the fraction ranges from 000000 to 999999. A decimal point must be used to separate the fraction from the rest.

- Use `type_name(fsp)` to define a column that supports fractional precision, where `type_name` can be TIME, DATETIME or TIMESTAMP. For example,

```
CREATE TABLE t1 (t TIME(3), dt DATETIME(6));
```

`fsp` must range from 0 to 6.

0 means there is no fractional part. If `fsp` is omitted, the default is 0.

- When inserting TIME, DATETIME or TIMESTAMP which contain a fractional part, if the number of digit of the fraction is too few, or too many, rounding might be needed in the situation. For example:

```
mysql> CREATE TABLE fractest( c1 TIME(2), c2 DATETIME(2), c3 TIMESTAMP
   ↪ (2) );
Query OK, 0 rows affected (0.33 sec)

mysql> INSERT INTO fractest VALUES
      > ('17:51:04.777', '2014-09-08 17:51:04.777', '2014-09-08
       ↪ 17:51:04.777');
Query OK, 1 row affected (0.03 sec)

mysql> SELECT * FROM fractest;
+-----+-----+-----+
| c1   | c2     | c3           |
+-----+-----+-----+
| 17:51:04.78 | 2014-09-08 17:51:04.78 | 2014-09-08 17:51:04.78 |
+-----+-----+-----+
1 row in set (0.00 sec)
```

11.5.3.4.4 Conversions between date and time types

Sometimes we need to make conversions between date and time types. But some conversions might lead to information loss. For example, DATE, DATETIME and TIMESTAMP values all have their own respective ranges. TIMESTAMP should be no earlier than the year 1970 in UTC time or no later than UTC time ‘2038-01-19 03:14:07’. Based on this rule, ‘1968-01-01’ is a valid date value of DATE or DATETIME, but becomes 0 when it is converted to TIMESTAMP.

The conversions of DATE:

- When DATE is converted to DATETIME or TIMESTAMP, a time-portion ‘00:00:00’ is added, because DATE does not contain any time information
- When DATE is converted to TIME, the result is ‘00:00:00’

Conversions of DATETIME or TIMESTAMP:

- When DATETIME or TIMESTAMP is converted to DATE, the time and fractional part is discarded. For example, ‘1999-12-31 23:59:59.499’ is converted to ‘1999-12-31’
- When DATETIME or TIMESTAMP is converted to TIME, the date-portion is discarded, because TIME does not contain any date information

When we convert TIME to other time and date formats, the date-portion is automatically specified as CURRENT_DATE(). The final converted result is a date that consists of TIME and CURRENT_DATE(). This is to say that if the value of TIME is beyond the range from ‘00:00:00’ to ‘23:59:59’, the converted date-portion does not indicate the current day.

When TIME is converted to DATE, the process is similar, and the time-portion is discarded.

Using the CAST() function can explicitly convert a value to a DATE type. For example:

```
date_col = CAST(datetime_col AS DATE)
```

Converting TIME and DATETIME to numeric format. For example:

mysql> SELECT CURTIME(), CURTIME()+0, CURTIME(3)+0;
+-----+-----+-----+
CURTIME() CURTIME()+0 CURTIME(3)+0
+-----+-----+-----+
09:28:00 92800 92800.887
+-----+-----+-----+
mysql> SELECT NOW(), NOW()+0, NOW(3)+0;
+-----+-----+-----+
NOW() NOW()+0 NOW(3)+0
+-----+-----+-----+
2012-08-15 09:28:00 20120815092800 20120815092800.889
+-----+-----+-----+

11.5.3.4.5 Two-digit year-portion contained in the date

The two-digit year-portion contained in date does not explicitly indicate the actual year and is ambiguous.

For DATETIME, DATE and TIMESTAMP types, TiDB follows the following rules to eliminate ambiguity:

- Values between 01 and 69 is converted to a value between 2001 and 2069
- Values between 70 and 99 is converted to a value between 1970 and 1999

These rules also apply to the YEAR type, with one exception:

When numeral 00 is inserted to YEAR(4), the result is 0000 rather than 2000.

If you want the result to be 2000, specify the value to be 2000.

The two-digit year-portion might not be properly calculated in some functions such MIN() and MAX(). For these functions, the four-digit format suites better.

11.5.3.5 String Types

TiDB supports all the MySQL string types, including CHAR, VARCHAR, BINARY, VARBINARY → , BLOB, TEXT, ENUM, and SET. For more information, see [String Types in MySQL](#).

11.5.3.5.1 Supported types

CHAR type

CHAR is a fixed length string. Values stored as CHAR are right-padded with spaces to the specified length. M represents the column-length in characters (not bytes). The range of M is 0 to 255:

`[NATIONAL] CHAR[(M)] [CHARACTER SET charset_name] [COLLATE collation_name]`

VARCHAR type

VARCHAR is a string of variable-length. M represents the maximum column length in characters (not bytes). The maximum size of VARCHAR cannot exceed 65,535 bytes. The maximum row length and the character set being used determine the VARCHAR length.

The space occupied by a single character might differ for different character sets. The following table shows the bytes consumed by a single character, and the range of the VARCHAR column length in each character set:

Character Set	Byte(s) per Character	Range of the Maximum VARCHAR Column Length
ascii	1	(0, 65535]
latin1	1	(0, 65535]
binary	1	(0, 65535]
utf8	3	(0, 21845]
utf8mb4	4	(0, 16383]

```
[NATIONAL] VARCHAR(M) [CHARACTER SET charset_name] [COLLATE collation_name]
```

TEXT type

TEXT is a string of variable-length. M represents the maximum column length in characters, ranging from 0 to 65,535. The maximum row length and the character set being used determine the TEXT length.

```
TEXT[(M)] [CHARACTER SET charset_name] [COLLATE collation_name]
```

TINYTEXT type

The TINYTEXT type is similar to the [TEXT type](#). The difference is that the maximum column length of TINYTEXT is 255.

```
TINYTEXT [CHARACTER SET charset_name] [COLLATE collation_name]
```

MEDIUMTEXT type

The MEDIUMTEXT type is similar to the [TEXT type](#). The difference is that the maximum column length of MEDIUMTEXT is 16,777,215.

```
MEDIUMTEXT [CHARACTER SET charset_name] [COLLATE collation_name]
```

LONGTEXT type

The LONGTEXT type is similar to the [TEXT type](#). The difference is that the maximum column length of LONGTEXT is 4,294,967,295.

```
LONGTEXT [CHARACTER SET charset_name] [COLLATE collation_name]
```

BINARY type

The BINARY type is similar to the [CHAR type](#). The difference is that BINARY stores binary byte strings.

```
BINARY(M)
```

VARBINARY type

The VARBINARY type is similar to the [VARCHAR type](#). The difference is that the VARBINARY stores binary byte strings.

```
VARBINARY(M)
```

BLOB type

BLOB is a large binary file. M represents the maximum column length in bytes, ranging from 0 to 65,535.

```
BLOB[(M)]
```

TINYBLOB type

The TINYBLOB type is similar to the [BLOB type](#). The difference is that the maximum column length of TINYBLOB is 255.

TINYBLOB

MEDIUMBLOB type

The MEDIUMBLOB type is similar to the [BLOB type](#). The difference is that the maximum column length of MEDIUMBLOB is 16,777,215.

MEDIUMBLOB

LONGBLOB type

The LONGBLOB type is similar to the [BLOB type](#). The difference is that the maximum column length of LONGBLOB is 4,294,967,295.

LONGBLOB

ENUM type

An ENUM is a string object with a value chosen from a list of permitted values that are enumerated explicitly in the column specification when the table is created. The syntax is:

```
ENUM('value1','value2',...) [CHARACTER SET charset_name] [COLLATE
    ↪ collation_name]
```

For example:

```
ENUM('apple', 'orange', 'pear')
```

The value of the ENUM data type is stored as numbers. Each value is converted to a number according the definition order. In the previous example, each string is mapped to a number:

Value	Number
NULL	NULL
”	0
‘apple’	1
‘orange’	2
‘pear’	3

For more information, see [the ENUM type in MySQL](#).

SET type

A SET is a string object that can have zero or more values, each of which must be chosen from a list of permitted values specified when the table is created. The syntax is:

```
SET('value1','value2',...) [CHARACTER SET charset_name] [COLLATE
↪ collation_name]

#### For example:
SET('1', '2') NOT NULL
```

In the example, any of the following values can be valid:

```
''
'1'
'2'
'1,2'
```

In TiDB, the values of the `SET` type is internally converted to `Int64`. The existence of each element is represented using a binary: 0 or 1. For a column specified as `SET('a','b'↪ ','c','d')`, the members have the following decimal and binary values.

Member	Decimal Value	Binary Value
‘a’	1	0001
‘b’	2	0010
‘c’	4	0100
‘d’	8	1000

In this case, for an element of ('a', 'c'), it is 0101 in binary.

For more information, see [the SET type in MySQL](#).

11.5.3.6 JSON Type

Warning:

This is still an experimental feature. It is **NOT** recommended that you use it in the production environment.

TiDB supports the `JSON` (JavaScript Object Notation) data type, which is useful for storing semi-structured data. The `JSON` data type provides the following advantages over storing `JSON`-format strings in a string column:

- Use the Binary format for serialization. The internal format permits quick read access to `JSON` document elements.

- Automatic validation of the JSON documents stored in JSON columns. Only valid documents can be stored.

JSON columns, like columns of other binary types, are not indexed directly, but you can index the fields in the JSON document in the form of generated column:

```
CREATE TABLE city (
    id INT PRIMARY KEY,
    detail JSON,
    population INT AS (JSON_EXTRACT(detail, '$.population')),
    INDEX index_name (population)
);
INSERT INTO city (id,detail) VALUES (1, '{"name": "Beijing", "population": 100});
SELECT id FROM city WHERE population >= 100;
```

For more information, see [JSON Functions](#) and [Generated Columns](#).

11.5.4 Functions and Operators

11.5.4.1 Function and Operator Reference

The usage of the functions and operators in TiDB is similar to MySQL. See [Functions and Operators in MySQL](#).

In SQL statements, expressions can be used on the ORDER BY and HAVING clauses of the SELECT statement, the WHERE clause of SELECT/DELETE/UPDATE statements, and SET statements.

You can write expressions using literals, column names, NULL, built-in functions, operators and so on. For expressions that TiDB supports pushing down to TiKV, see [List of Expressions for Pushdown](#).

11.5.4.2 Type Conversion in Expression Evaluation

TiDB behaves the same as MySQL: <https://dev.mysql.com/doc/refman/5.7/en/type-conversion.html>

11.5.4.3 Operators

This document describes the operators precedence, comparison functions and operators, logical operators, and assignment operators.

- Operator precedence
- Comparison functions and operators
- Logical operators
- Assignment operators

Name	Description
AND, &&	Logical AND
=	Assign a value (as part of a SET statement, or as part of the SET clause in an UPDATE statement)
:=	Assign a value
BETWEEN ... AND ...	Check whether a value is within a range of values
BINARY	Cast a string to a binary string
&	Bitwise AND
~	Bitwise inversion
	Bitwise OR
[^](https://dev.mysql.com/doc/refman/5.7/en/bitwise-functions.html#operator_bitwise-xor)	Bitwise XOR
CASE	Case operator
DIV	Integer division
/	Division operator
=	Equal operator
<=>	NULL-safe equal to operator
>	Greater than operator
>=	Greater than or equal operator
IS	Test a value against a boolean
IS NOT	Test a value against a boolean
IS NOT NULL	NOT NULL value test
IS NULL	NULL value test
->	Return value from JSON column after evaluating path; equivalent to JSON_EXTRACT()
->>	Return value from JSON column after evaluating path and unquoting the result; equivalent to JSON_UNQUOTE(JSON_EXTRACT())
<<	Left shift
<	Less than operator
<=	Less than or equal operator
LIKE	Simple pattern matching
-	Minus operator
%, MOD	Modulo operator
NOT, !	Negates value
NOT BETWEEN ... AND ...	Check whether a value is not within a range of values
!=, <>	Not equal operator
NOT LIKE	Negation of simple pattern matching
NOT REGEXP	Negation of REGEXP

Name	Description
<code> , OR</code>	Logical OR
<code>+</code>	Addition operator
<code>REGEXP</code>	Pattern matching using regular expressions
<code>>></code>	Right shift
<code>RLIKE</code>	Synonym for REGEXP
<code>SOUNDS LIKE</code>	Compare sounds
<code>*</code>	Multiplication operator
<code>-</code>	Change the sign of the argument
<code>XOR</code>	Logical XOR

11.5.4.3.1 Operator precedence

Operator precedences are shown in the following list, from highest precedence to the lowest. Operators that are shown together on a line have the same precedence.

```

INTERVAL
BINARY, COLLATE
!
-
- (unary minus), ~ (unary bit inversion)
^

*, /, DIV, %, MOD
-, +
<<, >>
&
|
=
(comparison), <=>, >=, >, <=, <, >>, !=, IS, LIKE, REGEXP, IN
BETWEEN, CASE, WHEN, THEN, ELSE
NOT
AND, &&
XOR
OR, ||
= (assignment), :=

```

For details, see [Operator Precedence](#).

11.5.4.3.2 Comparison functions and operators

Name	Description
<code>BETWEEN ... AND ...</code>	Check whether a value is within a range of values
<code>COALESCE()</code>	Return the first non-NULL argument
<code>=</code>	Equal operator
<code><=></code>	NULL-safe equal to operator
<code>></code>	Greater than operator

Name	Description
<code>>=</code>	Greater than or equal operator
<code>GREATEST()</code>	Return the largest argument
<code>IN()</code>	Check whether a value is within a set of values
<code>INTERVAL()</code>	Return the index of the argument that is less than the first argument
<code>IS</code>	Test a value against a boolean
<code>IS NOT</code>	Test a value against a boolean
<code>IS NOT NULL</code>	NOT NULL value test
<code>IS NULL</code>	NULL value test
<code>ISNULL()</code>	Test whether the argument is NULL
<code>LEAST()</code>	Return the smallest argument
<code><</code>	Less than operator
<code><=</code>	Less than or equal operator
<code>LIKE</code>	Simple pattern matching
<code>NOT BETWEEN ... AND ...</code>	Check whether a value is not within a range of values
<code>!=, <></code>	Not equal operator
<code>NOT IN()</code>	Check whether a value is not within a set of values
<code>NOT LIKE</code>	Negation of simple pattern matching
<code>STRCMP()</code>	Compare two strings

For details, see [Comparison Functions and Operators](#).

11.5.4.3.3 Logical operators

Name	Description
<code>AND, &&</code>	Logical AND
<code>NOT, !</code>	Negates value
<code> , OR</code>	Logical OR
<code>XOR</code>	Logical XOR

For details, see [MySQL Handling of GROUP BY](#).

11.5.4.3.4 Assignment operators

Name	Description
<code>=</code>	Assign a value (as part of a <code>SET</code> statement, or as part of the <code>SET</code> clause in an <code>UPDATE</code> statement)
<code>:=</code>	Assign a value

For details, see [Detection of Functional Dependence](#).

11.5.4.4 Control Flow Functions

TiDB supports all of the [control flow functions](#) available in MySQL 5.7.

Name	Description
<code>CASE</code>	Case operator
<code>IF()</code>	If/else construct
<code>IFNULL()</code>	Null if/else construct
<code>NULLIF()</code>	Return NULL if expr1 = expr2

11.5.4.5 String Functions

TiDB supports most of the [string functions](#) available in MySQL 5.7 and some of the [functions](#) available in Oracle 21.

11.5.4.5.1 Supported functions

Name	Description
<code>ASCII()</code>	Return numeric value of left-most character
<code>BIN()</code>	Return a string containing binary representation of a number
<code>BIT_LENGTH()</code>	Return length of argument in bits
<code>CHAR()</code>	Return the character for each integer passed
<code>CHAR_LENGTH()</code>	Return number of characters in argument
<code>CHARACTER_LENGTH()</code>	Synonym for <code>CHAR_LENGTH()</code>
<code>CONCAT()</code>	Return concatenated string
<code>CONCAT_WS()</code>	Return concatenate with separator
<code>ELT()</code>	Return string at index number
<code>EXPORT_SET()</code>	Return a string such that for every bit set in the value bits, you get an on string and for every unset bit, you get an off string
<code>FIELD()</code>	Return the index (position) of the first argument in the subsequent arguments
<code>FIND_IN_SET()</code>	Return the index position of the first argument within the second argument
<code>FORMAT()</code>	Return a number formatted to specified number of decimal places
<code>FROM_BASE64()</code>	Decode to a base-64 string and return result
<code>HEX()</code>	Return a hexadecimal representation of a decimal or string value

Name	Description
INSERT()	Insert a substring at the specified position up to the specified number of characters
INSTR()	Return the index of the first occurrence of substring
LCASE()	Synonym for LOWER()
LEFT()	Return the leftmost number of characters as specified
LENGTH()	Return the length of a string in bytes
LIKE	Simple pattern matching
LOCATE()	Return the position of the first occurrence of substring
LOWER()	Return the argument in lowercase
LPAD()	Return the string argument, left-padded with the specified string
LTRIM()	Remove leading spaces
MAKE_SET()	Return a set of comma-separated strings that have the corresponding bit in bits set
MID()	Return a substring starting from the specified position
NOT LIKE	Negation of simple pattern matching
NOT REGEXP	Negation of REGEXP
OCT()	Return a string containing octal representation of a number
OCTET_LENGTH()	Synonym for LENGTH()
ORD()	Return character code for leftmost character of the argument
POSITION()	Synonym for LOCATE()
QUOTE()	Escape the argument for use in an SQL statement
REGEXP	Pattern matching using regular expressions
REPEAT()	Repeat a string the specified number of times
REPLACE()	Replace occurrences of a specified string
REVERSE()	Reverse the characters in a string
RIGHT()	Return the specified rightmost number of characters
RLIKE	Synonym for REGEXP
RPAD()	Append string the specified number of times
RTRIM()	Remove trailing spaces
SPACE()	Return a string of the specified number of spaces

Name	Description
STRCMP()	Compare two strings
SUBSTR()	Return the substring as specified
SUBSTRING()	Return the substring as specified
SUBSTRING_INDEX()	Return a substring from a string before the specified number of occurrences of the delimiter
TO_BASE64()	Return the argument converted to a base-64 string
TRANSLATE()	Replace all occurrences of characters by other characters in a string. It does not treat empty strings as NULL as Oracle does.
TRIM()	Remove leading and trailing spaces
UCASE()	Synonym for UPPER()
UNHEX()	Return a string containing hex representation of a number
UPPER()	Convert to uppercase

11.5.4.5.2 Unsupported functions

- LOAD_FILE()
- MATCH
- SOUNDEX()
- SOUNDS LIKE
- WEIGHT_STRING()

11.5.4.6 Numeric Functions and Operators

TiDB supports all of the [numeric functions and operators](#) available in MySQL 5.7.

11.5.4.6.1 Arithmetic operators

Name	Description
+	Addition operator
-	Minus operator
*	Multiplication operator
/	Division operator
DIV	Integer division
lstinlineMOD!	Modulo operator
-	Change the sign of the argument

11.5.4.6.2 Mathematical functions

Name	Description
POW()	Return the argument raised to the specified power
POWER()	Return the argument raised to the specified power
EXP()	Raise to the power of
SQRT()	Return the square root of the argument
LN()	Return the natural logarithm of the argument
LOG()	Return the natural logarithm of the first argument
LOG2()	Return the base-2 logarithm of the argument
LOG10()	Return the base-10 logarithm of the argument
PI()	Return the value of pi
TAN()	Return the tangent of the argument
COT()	Return the cotangent
SIN()	Return the sine of the argument
COS()	Return the cosine
ATAN()	Return the arc tangent
ATAN2(), ATAN()	Return the arc tangent of the two arguments
ASIN()	Return the arc sine
ACOS()	Return the arc cosine
RADIANS()	Return argument converted to radians
DEGREES()	Convert radians to degrees
MOD()	Return the remainder
ABS()	Return the absolute value
CEIL()	Return the smallest integer value not less than the argument
CEILING()	Return the smallest integer value not less than the argument
FLOOR()	Return the largest integer value not greater than the argument
ROUND()	Round the argument
RAND()	Return a random floating-point value
SIGN()	Return the sign of the argument
CONV()	Convert numbers between different number bases
TRUNCATE()	Truncate to specified number of decimal places
CRC32()	Compute a cyclic redundancy check value

11.5.4.7 Date and Time Functions

TiDB supports all of the [date and time functions](#) available in MySQL 5.7.

Date/Time functions:

Name	Description
ADDDATE()	Add time values (intervals) to a date value
ADDTIME()	Add time
CONVERT_TZ()	Convert from one time zone to another

Name	Description
CURDATE()	Return the current date
CURRENT_DATE(), CURRENT_DATE	Synonyms for CURDATE()
CURRENT_TIME(), CURRENT_TIME	Synonyms for CURTIME()
CURRENT_TIMESTAMP(), CURRENT_TIMESTAMP	Synonyms for NOW()
CURTIME()	Return the current time
DATE()	Extract the date part of a date or datetime expression
DATE_ADD()	Add time values (intervals) to a date value
DATE_FORMAT()	Format date as specified
DATE_SUB()	Subtract a time value (interval) from a date
DATEDIFF()	Subtract two dates
DAY()	Synonym for DAYOFMONTH()
DAYNAME()	Return the name of the weekday
DAYOFMONTH()	Return the day of the month (0-31)
DAYOFWEEK()	Return the weekday index of the argument
DAYOFYEAR()	Return the day of the year (1-366)
EXTRACT()	Extract part of a date
FROM_DAYS()	Convert a day number to a date
FROM_UNIXTIME()	Format Unix timestamp as a date
GET_FORMAT()	Return a date format string
HOUR()	Extract the hour
LAST_DAY	Return the last day of the month for the argument
LOCALTIME(), LOCALTIME	Synonym for NOW()
LOCALTIMESTAMP, LOCALTIMESTAMP()	Synonym for NOW()
MAKEDATE()	Create a date from the year and day of year
MAKETIME()	Create time from hour, minute, second
MICROSECOND()	Return the microseconds from argument
MINUTE()	Return the minute from the argument
MONTH()	Return the month from the date passed
MONTHNAME()	Return the name of the month
NOW()	Return the current date and time
PERIOD_ADD()	Add a period to a year-month
PERIOD_DIFF()	Return the number of months between periods
QUARTER()	Return the quarter from a date argument
SEC_TO_TIME()	Converts seconds to 'HH:MM:SS' format
SECOND()	Return the second (0-59)
STR_TO_DATE()	Convert a string to a date

Name	Description
SUBDATE()	Synonym for DATE_SUB() when invoked with three arguments
SUBTIME()	Subtract times
SYSDATE()	Return the time at which the function executes
TIME()	Extract the time portion of the expression passed
TIME_FORMAT()	Format as time
TIME_TO_SEC()	Return the argument converted to seconds
TIMEDIFF()	Subtract time
TIMESTAMP()	With a single argument, this function returns the date or datetime expression; with two arguments, the sum of the arguments
TIMESTAMPADD()	Add an interval to a datetime expression
TIMESTAMPDIFF()	Subtract an interval from a datetime expression
TO_DAYS()	Return the date argument converted to days
TO_SECONDS()	Return the date or datetime argument converted to seconds since Year 0
UNIX_TIMESTAMP()	Return a Unix timestamp
UTC_DATE()	Return the current UTC date
UTC_TIME()	Return the current UTC time
UTC_TIMESTAMP()	Return the current UTC date and time
WEEK()	Return the week number
WEEKDAY()	Return the weekday index
WEEKOFYEAR()	Return the calendar week of the date (1-53)
YEAR()	Return the year
YEARWEEK()	Return the year and week

For details, see [Date and Time Functions](#).

11.5.4.7.1 Related system variables

The `default_week_format` variable affects the `WEEK()` function.

11.5.4.8 Bit Functions and Operators

TiDB supports all of the [bit functions and operators](#) available in MySQL 5.7.

Bit functions and operators:

Name	Description
<code>BIT_COUNT</code>	Return the number of bits that are set as 1
<code>&</code>	Bitwise AND
<code>~</code>	Bitwise inversion
<code> </code>	Bitwise OR
<code>[^](https://BitwiseXOR.com/doc/refman/5.7/en/bit-functions.html#operator_bitwise-xor)</code>	Bitwise XOR
<code><<</code>	Left shift
<code>>></code>	Right shift

11.5.4.9 Cast Functions and Operators

TiDB supports all of the [cast functions and operators](#) available in MySQL 5.7.

Name	Description
<code>BINARY</code>	Cast a string to a binary string
<code>CAST()</code>	Cast a value as a certain type
<code>CONVERT()</code>	Cast a value as a certain type

Cast functions and operators enable conversion of values from one data type to another.

11.5.4.10 Encryption and Compression Functions

TiDB supports most of the [encryption and compression functions](#) available in MySQL 5.7.

11.5.4.10.1 Supported functions

Name	Description
<code>MD5()</code>	Calculate MD5 checksum
<code>PASSWORD()</code>	Calculate and return a password string
<code>RANDOM_BYTES()</code>	Return a random byte vector
<code>SHA1(), SHA()</code>	Calculate an SHA-1 160-bit checksum
<code>SHA2()</code>	Calculate an SHA-2 checksum
<code>AES_DECRYPT()</code>	Decrypt using AES
<code>AES_ENCRYPT()</code>	Encrypt using AES
<code>COMPRESS()</code>	Return result as a binary string
<code>UNCOMPRESS()</code>	Uncompress a string compressed
<code>UNCOMPRESSED_LENGTH()</code>	Return the length of a string before compression

Name	Description
<code>CREATE_ASYMMETRIC_PRIV_KEY()</code>	Create private key
<code>CREATE_ASYMMETRIC_PUB_KEY()</code>	Create public key
<code>CREATE_DH_PARAMETERS()</code>	Generate shared DH secret
<code>CREATE_DIGEST()</code>	Generate digest from string
<code>ASYMMETRIC_DECRYPT()</code>	Decrypt ciphertext using private or public key
<code>ASYMMETRIC_DERIVE()</code>	Derive symmetric key from asymmetric keys
<code>ASYMMETRIC_ENCRYPT()</code>	Encrypt cleartext using private or public key
<code>ASYMMETRIC_SIGN()</code>	Generate signature from digest
<code>ASYMMETRIC_VERIFY()</code>	Verify that signature matches digest

11.5.4.10.2 Related system variables

The `block_encryption_mode` variable sets the encryption mode that is used for `AES_ENCRYPT()` and `AES_DECRYPT()`.

11.5.4.10.3 Unsupported functions

- `DES_DECRYPT()`, `DES_ENCRYPT()`, `OLD_PASSWORD()`, `ENCRYPT()`: these functions were deprecated in MySQL 5.7 and removed in 8.0.
- `VALIDATE_PASSWORD_STRENGTH()`
- Functions only available in MySQL Enterprise Issue #2632

11.5.4.11 Information Functions

TiDB supports most of the [information functions](#) available in MySQL 5.7.

11.5.4.11.1 Supported functions

Name	Description
<code>BENCHMARK(→ ()</code>	Execute an expression in a loop
<code>CONNECTION_ID(→ ()</code>	Return the connection ID (thread ID) for the connection
<code>CURRENT_USER(→ (), CURRENT_USER(→ ()</code>	Return the authenticated user name and host name

Name	Description
DATABASE	Return the → () default (current) database name
FOUND_ROWS	For a SELECT → () with a LIMIT clause, the number of the rows that are returned if there is no LIMIT clause
LAST_INSERT_ID	Return the → () value of the AUTOINCREMENT → column for the last INSERT
ROW_COUNT	The number → () of rows affected
SCHEMA	Synonym for → () DATABASE()
SESSION_USER	Synonym for → () USER()
SYSTEM_USER	Synonym for → () USER()
USER()	Return the user name and host name provided by the client
VERSION	Return a → () string that indicates the MySQL server version

11.5.4.11.2 Unsupported functions

- CHARSET()

- COERCIBILITY()
- COLLATION()

11.5.4.12 JSON Functions

Warning:

This is still an experimental feature. It is **NOT** recommended that you use it in the production environment.

TiDB supports most of the JSON functions that shipped with the GA release of MySQL 5.7. Additional JSON functions were added to MySQL 5.7 after its release, and not all are available in TiDB (see [unsupported functions](#)).

11.5.4.12.1 Functions that create JSON values

Function Name	Description
JSON_ARRAY([val[, val] ...])	Evaluates a (possibly empty) list of values and returns a JSON array containing those values
JSON_OBJECT(key, val[, key, val ...])	Evaluates a (possibly empty) list of key-value pairs and returns a JSON object containing those pairs
JSON_QUOTE(string)	Returns a string as a JSON value with quotes

11.5.4.12.2 Functions that search JSON values

Function Name	Description
<code>JSON_CONTAINS(target, candidate[, path])</code>	Indicates by returning 1 or 0 whether a given candidate JSON document is contained within a target JSON document
<code>JSON_CONTAINS_PATH(json_doc, one_or_all, path[, path] ...)</code>	Returns 0 or 1 to indicate whether a JSON document contains data at a given path or paths
<code>JSON_EXTRACT(json_doc, path[, path] ...)</code>	Returns data from a JSON document, selected from the parts of the document matched by the <code>path</code> arguments

Function Name	Description
->	Returns the value from a JSON column after evaluating path; an alias for JSON_EXTRACT ↳ (doc, ↳ path_literal ↳)
->>	Returns the value from a JSON column after evaluating path and unquoting the result; an alias for JSON_UNQUOTE ↳ (↳ JSON_EXTRACT ↳ (doc, ↳ path_literal ↳))
JSON_KEYS(json_doc[, path])	Returns the keys from the top-level value of a JSON object as a JSON array, or, if a path argument is given, the top-level keys from the selected path

Function Name	Description
JSON_SEARCH(json_doc, one_or_all, search_string)	Search a JSON document for one or all matches of a string

11.5.4.12.3 Functions that modify JSON values

Function Name	Description
JSON_APPEND(json_doc, path, value)	An alias to JSON_ARRAY_APPEND ↔
JSON_ARRAY_APPEND(json_doc, path, value)	Appends a value to the end of a JSON array at a specified path
JSON_ARRAY_INSERT(json_doc, path, val[, path, val] ...)	Inserts an array into the json document and returns the modified document
JSON_INSERT(json_doc, path, val[, path, val] ...)	Inserts data into a JSON document and returns the result
JSON.Merge(json_doc, json_doc[, json_doc] ...)	A deprecated alias for JSON.Merge_Preserve →

Function Name	Description
JSON_MERGE_PRESERVE(json_doc[, json_doc] ...)	Merges two or more JSON documents and returns the merged result
JSON_REMOVE(json_doc, path[, path] ...)	Removes data from a JSON document and returns the result
JSON_REPLACE(json_doc, path, val[, path, val] ...)	Replaces existing values in a JSON document and returns the result
JSON_SET(json_doc, path, val[, path, val] ...)	Inserts or updates data in a JSON document and returns the result
JSON_UNQUOTE(json_val)	Unquotes a JSON value and returns the result as a string
JSON_ARRAY_APPEND(json_doc, path, val[, path, val] ...)	Appends values to the end of the indicated arrays within a JSON document and returns the result

Function Name	Description
<code>JSON_ARRAY_INSERT(json_doc, path, val[, path, val] ...)</code>	Insert values into the specified location of a JSON document and returns the result

11.5.4.12.4 Functions that return JSON value attributes

Function Name	Description
<code>JSON_DEPTH(json_doc)</code>	Returns the maximum depth of a JSON document
<code>JSON_LENGTH(json_doc[, path])</code>	Returns the length of a JSON document, or, if a path argument is given, the length of the value within the path
<code>JSON_TYPE(json_val)</code>	Returns a string indicating the type of a JSON value
<code>JSON_VALID(json_doc)</code>	Checks if a json_doc is valid JSON. Useful for checking a column before converting it to the json type.

11.5.4.12.5 Utility Functions

Function Name	Description
[JSON_STORAGE_SIZE(json_doc)][JSON_STORAGE_SIZE]	approximate size of bytes required to store the json value. As the size does not account for TiKV using compression, the output of this function is not strictly compatible with MySQL.

11.5.4.12.6 Aggregate Functions

Function Name	Description
[JSON_OBJECTAGG(key, value)][JSON_OBJECTAGG]	Provides an aggregation of values for a given key.

11.5.4.12.7 Unsupported functions

The following JSON functions are unsupported in TiDB. You can track the progress in adding them in [TiDB #7546](#):

- [JSON_MERGE_PATCH](#)
- [JSON_PRETTY](#)
- [JSON_ARRAYAGG](#)

11.5.4.12.8 See also

- [JSON Function Reference](#)
- [JSON Data Type](#)

11.5.4.13 Aggregate (GROUP BY) Functions

This document describes details about the supported aggregate functions in TiDB.

11.5.4.13.1 Supported aggregate functions

This section describes the supported MySQL GROUP BY aggregate functions in TiDB.

Name	Description
COUNT()	Return a count of the number of rows returned
COUNT(DISTINCT)	Return the count of a number of different values
SUM()	Return the sum
AVG()	Return the average value of the argument
MAX()	Return the maximum value
MIN()	Return the minimum value
GROUP_CONCAT()	Return a concatenated string
VARIANCE() , VAR_POP()	Return the population standard variance
STD() , STDDEV() , STDDEV_POP	Return the population standard deviation
VAR_SAMP()	Return the sample variance
STDDEV_SAMP()	Return the sample standard deviation
JSON_OBJECTAGG(key, value)	Return the result set as a single JSON object containing key-value pairs

- Unless otherwise stated, group functions ignore NULL values.
- If you use a group function in a statement containing no GROUP BY clause, it is equivalent to grouping on all rows.

In addition, TiDB also provides the following aggregate functions:

- [APPROX_PERCENTILE\(expr, constant_integer_expr\)](#)

This function returns the percentile of `expr`. The `constant_integer_expr` argument indicates the percentage value which is a constant integer in the range of [1, 100]. A percentile Pk (`k` represents percentage) indicates that there are at least `k%` values in the data set that are less than or equal to Pk.

This function only supports the `numeric type` and the `date and time type` as the returned type of `expr`. For other returned types, `APPROX_PERCENTILE` only returns NULL.

The following example shows how to calculate the fiftieth percentile of a INT column:

```
drop table if exists t;
create table t(a int);
insert into t values(1), (2), (3);

select approx_percentile(a, 50) from t;
```

```
+-----+
| approx_percentile(a, 50) |
+-----+
|          2 |
+-----+
1 row in set (0.00 sec)
```

11.5.4.13.2 GROUP BY modifiers

TiDB does not currently support GROUP BY modifiers such as WITH ROLLUP. We plan to add support in the future. See [TiDB #4250](#).

11.5.4.13.3 SQL mode support

TiDB supports the SQL Mode ONLY_FULL_GROUP_BY, and when enabled TiDB will refuse queries with ambiguous non-aggregated columns. For example, this query is illegal with ONLY_FULL_GROUP_BY enabled because the non-aggregated column “b” in the SELECT list does not appear in the GROUP BY statement:

```
drop table if exists t;
create table t(a bigint, b bigint, c bigint);
insert into t values(1, 2, 3), (2, 2, 3), (3, 2, 3);

mysql> select a, b, sum(c) from t group by a;
+---+---+-----+
| a | b | sum(c) |
+---+---+-----+
| 1 | 2 |      3 |
| 2 | 2 |      3 |
| 3 | 2 |      3 |
+---+---+-----+
3 rows in set (0.01 sec)

mysql> set sql_mode = 'ONLY_FULL_GROUP_BY';
Query OK, 0 rows affected (0.00 sec)

mysql> select a, b, sum(c) from t group by a;
```

```
ERROR 1055 (42000): Expression #2 of SELECT list is not in GROUP BY clause
  ↪ and contains nonaggregated column 'b' which is not functionally
  ↪ dependent on columns in GROUP BY clause; this is incompatible with
  ↪ sql_mode=only_full_group_by
```

TiDB currently enables the `ONLY_FULL_GROUP_BY` mode by default.

Differences from MySQL

The current implementation of `ONLY_FULL_GROUP_BY` is less strict than that in MySQL 5.7. For example, suppose that we execute the following query, expecting the results to be ordered by “c”:

```
drop table if exists t;
create table t(a bigint, b bigint, c bigint);
insert into t values(1, 2, 1), (1, 2, 2), (1, 3, 1), (1, 3, 2);
select distinct a, b from t order by c;
```

To order the result, duplicates must be eliminated first. But to do so, which row should we keep? This choice influences the retained value of “c”, which in turn influences ordering and makes it arbitrary as well.

In MySQL, a query that has `DISTINCT` and `ORDER BY` is rejected as invalid if any `ORDER BY` expression does not satisfy at least one of these conditions:

- The expression is equal to one in the `SELECT` list
- All columns referenced by the expression and belonging to the query’s selected tables are elements of the `SELECT` list

But in TiDB, the above query is legal, for more information see [#4254](#).

Another TiDB extension to standard SQL permits references in the `HAVING` clause to aliased expressions in the `SELECT` list. For example, the following query returns “name” values that occur only once in table “orders”:

```
select name, count(name) from orders
group by name
having count(name) = 1;
```

The TiDB extension permits the use of an alias in the `HAVING` clause for the aggregated column:

```
select name, count(name) as c from orders
group by name
having c = 1;
```

Standard SQL permits only column expressions in `GROUP BY` clauses, so a statement such as this is invalid because “`FLOOR(value/100)`” is a noncolumn expression:

```
select id, floor(value/100)
from tbl_name
group by id, floor(value/100);
```

TiDB extends standard SQL to permit noncolumn expressions in GROUP BY clauses and considers the preceding statement valid.

Standard SQL also does not permit aliases in GROUP BY clauses. TiDB extends standard SQL to permit aliases, so another way to write the query is as follows:

```
select id, floor(value/100) as val
from tbl_name
group by id, val;
```

11.5.4.13.4 Unsupported aggregate functions

The following aggregate functions are currently unsupported in TiDB. You can track our progress in [TiDB #7623](#):

- JSON_ARRAYAGG

11.5.4.13.5 Related system variables

The `group_concat_max_len` variable sets the maximum number of items for the `GROUP_CONCAT()` function.

11.5.4.14 Window Functions

The usage of window functions in TiDB is similar to that in MySQL 8.0. For details, see [MySQL Window Functions](#).

Because window functions reserve additional words in the parser, TiDB provides an option to disable window functions. If you receive errors parsing SQL statements after upgrading, try setting `tidb_enable_window_function=0`.

TiDB supports the following window functions:

Function name	Feature description
<code>CUME_DIST()</code>	Returns the cumulative distribution of a value within a group of values.
<code>DENSE_RANK()</code>	Returns the rank of the current row within the partition, and the rank is without gaps.
<code>FIRST_VALUE()</code>	Returns the expression value of the first row in the current window.

Function name	Feature description
<code>LAG()</code>	Returns the expression value from the row that precedes the current row by N rows within the partition.
<code>LAST_VALUE()</code>	Returns the expression value of the last row in the current window.
<code>LEAD()</code>	Returns the expression value from the row that follows the current row by N rows within the partition.
<code>NTH_VALUE()</code>	Returns the expression value from the N-th row of the current window.
<code>NTILE()</code>	Divides a partition into N buckets, assigns the bucket number to each row in the partition, and returns the bucket number of the current row within the partition.
<code>PERCENT_RANK()</code>	Returns the percentage of partition values that are less than the value in the current row.
<code>RANK()</code>	Returns the rank of the current row within the partition. The rank may be with gaps.
<code>ROW_NUMBER()</code>	Returns the number of the current row in the partition.

11.5.4.15 Miscellaneous Functions

TiDB supports most of the [miscellaneous functions](#) available in MySQL 5.7.

11.5.4.15.1 Supported functions

Name	Description
<code>ANY_VALUE()</code>	Suppress <code>ONLY_FULL_GROUP_BY</code> value rejection
<code>DEFAULT()</code>	Returns the default value for a table column
<code>INET_ATON()</code>	Return the numeric value of an IP address
<code>INET_NTOA()</code>	Return the IP address from a numeric value
<code>INET6_ATON()</code>	Return the numeric value of an IPv6 address
<code>INET6_NTOA()</code>	Return the IPv6 address from a numeric value
<code>IS_IPV4()</code>	Whether argument is an IPv4 address
<code>IS_IPV4_COMPAT()</code>	Whether argument is an IPv4-compatible address
<code>IS_IPV4_MAPPED()</code>	Whether argument is an IPv4-mapped address
<code>IS_IPV6()</code>	Whether argument is an IPv6 address
<code>NAME_CONST()</code>	Can be used to rename a column name
<code>SLEEP()</code>	Sleep for a number of seconds

Name	Description
UUID()	Return a Universal Unique Identifier (UUID)
VALUES()	Defines the values to be used during an INSERT

11.5.4.15.2 Unsupported functions

Name	Description
GET_LOCK	Get a named lock TiDB #10929 ↪ ()
RELEASE_LOCK	Releases the named lock TiDB #10929 ↪ ()
UUID_SHORT	Provides a UUID that is unique given certain assumptions not present in ↪ () TiDB #4620
MASTER_WAIT	Refers to MySQL replication ↪ ()

11.5.4.16 Precision Math

The precision math support in TiDB is consistent with MySQL. For more information, see [Precision Math in MySQL](#).

11.5.4.16.1 Numeric types

The scope of precision math for exact-value operations includes the exact-value data types (integer and DECIMAL types) and exact-value numeric literals. Approximate-value data types and numeric literals are handled as floating-point numbers.

Exact-value numeric literals have an integer part or fractional part, or both. They may be signed. Examples: 1, .2, 3.4, -5, -6.78, +9.10.

Approximate-value numeric literals are represented in scientific notation (power-of-10) with a mantissa and exponent. Either or both parts may be signed. Examples: 1.2E3, 1.2E-3, -1.2E3, -1.2E-3.

Two numbers that look similar might be treated differently. For example, 2.34 is an exact-value (fixed-point) number, whereas 2.34E0 is an approximate-value (floating-point) number.

The DECIMAL data type is a fixed-point type and the calculations are exact. The FLOAT and DOUBLE data types are floating-point types and calculations are approximate.

11.5.4.16.2 DECIMAL data type characteristics

This section discusses the following topics of the characteristics of the DECIMAL data type (and its synonyms):

- Maximum number of digits
- Storage format
- Storage requirements

The declaration syntax for a DECIMAL column is `DECIMAL(M,D)`. The ranges of values for the arguments are as follows:

- M is the maximum number of digits (the precision). $1 \leq M \leq 65$.
- D is the number of digits to the right of the decimal point (the scale). $1 \leq D \leq 30$ and D must be no larger than M.

The maximum value of 65 for M means that calculations on DECIMAL values are accurate up to 65 digits. This limit of 65 digits of precision also applies to exact-value numeric literals.

Values for DECIMAL columns are stored using a binary format that packs 9 decimal digits into 4 bytes. The storage requirements for the integer and fractional parts of each value are determined separately. Each multiple of 9 digits requires 4 bytes, and any remaining digits left over require some fraction of 4 bytes. The storage required for remaining digits is given by the following table.

Leftover Digits	Number of Bytes
0	0
1–2	1
3–4	2
5–6	3
7–9	4

For example, a `DECIMAL(18,9)` column has 9 digits on each side of the decimal point, so the integer part and the fractional part each require 4 bytes. A `DECIMAL(20,6)` column has 14 integer digits and 6 fractional digits. The integer digits require 4 bytes for 9 of the digits and 3 bytes for the remaining 5 digits. The 6 fractional digits require 3 bytes.

DECIMAL columns do not store a leading + character or – character or leading 0 digits. If you insert `+0003.1` into a `DECIMAL(5,1)` column, it is stored as `3.1`. For negative numbers, a literal – character is not stored.

DECIMAL columns do not permit values larger than the range implied by the column definition. For example, a `DECIMAL(3,0)` column supports a range of -999 to 999. A `DECIMAL(M,D)` column permits at most $M - D$ digits to the left of the decimal point.

For more information about the internal format of the DECIMAL values, see [mydecimal.go](#) in TiDB source code.

11.5.4.16.3 Expression handling

For expressions with precision math, TiDB uses the exact-value numbers as given whenever possible. For example, numbers in comparisons are used exactly as given without a change in value. In strict SQL mode, if you add an exact data type into a column, a number is inserted with its exact value if it is within the column range. When retrieved, the value is the same as what is inserted. If strict SQL mode is not enabled, truncation for INSERT is permitted in TiDB.

How to handle a numeric expression depends on the values of the expression:

- If the expression contains any approximate values, the result is approximate. TiDB evaluates the expression using floating-point arithmetic.
- If the expression contains no approximate values are present, which means only exact values are contained, and if any exact value contains a fractional part, the expression is evaluated using DECIMAL exact arithmetic and has a precision of 65 digits.
- Otherwise, the expression contains only integer values. The expression is exact. TiDB evaluates the expression using integer arithmetic and has a precision the same as BIGINT (64 bits).

If a numeric expression contains strings, the strings are converted to double-precision floating-point values and the result of the expression is approximate.

Inserts into numeric columns are affected by the SQL mode. The following discussions mention strict mode and `ERROR_FOR_DIVISION_BY_ZERO`. To turn on all the restrictions, you can simply use the `TRADITIONAL` mode, which includes both strict mode values and `ERROR_FOR_DIVISION_BY_ZERO`:

```
SET sql_mode = 'TRADITIONAL';
```

If a number is inserted into an exact type column (DECIMAL or integer), it is inserted with its exact value if it is within the column range. For this number:

- If the value has too many digits in the fractional part, rounding occurs and a warning is generated.
- If the value has too many digits in the integer part, it is too large and is handled as follows:
 - If strict mode is not enabled, the value is truncated to the nearest legal value and a warning is generated.
 - If strict mode is enabled, an overflow error occurs.

To insert strings into numeric columns, TiDB handles the conversion from string to number as follows if the string has nonnumeric contents:

- In strict mode, a string (including an empty string) that does not begin with a number cannot be used as a number. An error, or a warning occurs.

- A string that begins with a number can be converted, but the trailing nonnumeric portion is truncated. In strict mode, if the truncated portion contains anything other than spaces, an error, or a warning occurs.

By default, the result of the division by 0 is NULL and no warning. By setting the SQL mode appropriately, division by 0 can be restricted. If you enable the `ERROR_FOR_DIVISION_BY_ZERO` SQL mode, TiDB handles division by 0 differently:

- In strict mode, inserts and updates are prohibited, and an error occurs.
- If it's not in the strict mode, a warning occurs.

In the following SQL statement:

```
INSERT INTO t SET i = 1/0;
```

The following results are returned in different SQL modes:

sql_mode	Value	Result
"		No warning, no error; i is set to NULL.
strict		No warning, no error; i is set to NULL.
<code>ERROR_FOR_DIVISION_BY_ZERO</code>		Warning, no error; i is set to NULL.
strict, <code>ERROR_FOR_DIVISION_BY_ZERO</code>		Error; no row is inserted.

11.5.4.16.4 Rounding behavior

The result of the `ROUND()` function depends on whether its argument is exact or approximate:

- For exact-value numbers, the `ROUND()` function uses the “round half up” rule.
- For approximate-value numbers, the results in TiDB differs from that in MySQL:

```
TiDB > SELECT ROUND(2.5), ROUND(25E-1);
+-----+
| ROUND(2.5) | ROUND(25E-1) |
+-----+
|      3     |       3      |
+-----+
1 row in set (0.00 sec)
```

For inserts into a DECIMAL or integer column, the rounding uses `round half away from zero`.

```

TiDB > CREATE TABLE t (d DECIMAL(10,0));
Query OK, 0 rows affected (0.01 sec)

TiDB > INSERT INTO t VALUES(2.5),(2.5E0);
Query OK, 2 rows affected, 2 warnings (0.00 sec)

TiDB > SELECT d FROM t;
+-----+
| d   |
+-----+
| 3  |
| 3  |
+-----+
2 rows in set (0.00 sec)

```

11.5.4.17 Set Operations

TiDB supports three set operations using the UNION, EXCEPT, and INTERSECT operators. The smallest unit of a set is a **SELECT statement**.

11.5.4.17.1 UNION operator

In mathematics, the union of two sets A and B consists of all elements that are in A or in B. For example:

```

select 1 union select 2;
+---+
| 1 |
+---+
| 2 |
| 1 |
+---+
2 rows in set (0.00 sec)

```

TiDB supports both UNION DISTINCT and UNION ALL operators. UNION DISTINCT removes duplicate records from the result set, while UNION ALL keeps all records including duplicates. UNION DISTINCT is used by default in TiDB.

```

create table t1 (a int);
create table t2 (a int);
insert into t1 values (1),(2);
insert into t2 values (1),(3);

```

Examples for UNION DISTINCT and UNION ALL queries are respectively as follows:

```
select * from t1 union distinct select * from t2;
+---+
| a |
+---+
| 1 |
| 2 |
| 3 |
+---+
3 rows in set (0.00 sec)
select * from t1 union all select * from t2;
+---+
| a |
+---+
| 1 |
| 2 |
| 1 |
| 3 |
+---+
4 rows in set (0.00 sec)
```

11.5.4.17.2 EXCEPT operator

If A and B are two sets, EXCEPT returns the difference set of A and B which consists of elements that are in A but not in B.

```
select * from t1 except select * from t2;
+---+
| a |
+---+
| 2 |
+---+
1 rows in set (0.00 sec)
```

EXCEPT ALL operator is not yet supported.

11.5.4.17.3 INTERSECT operator

In mathematics, the intersection of two sets A and B consists of all elements that are both in A and B, and no other elements.

```
select * from t1 intersect select * from t2;
+---+
| a |
+---+
| 1 |
```

```
+---+
1 rows in set (0.00 sec)
```

INTERSECT ALL operator is not yet supported. INTERSECT operator has higher precedence over EXCEPT and UNION operators.

```
select * from t1 union all select * from t1 intersect select * from t2;
+---+
| a |
+---+
| 1 |
| 1 |
| 2 |
+---+
3 rows in set (0.00 sec)
```

11.5.4.17.4 Parentheses

TiDB supports using parentheses to specify the precedence of set operations. Expressions in parentheses are processed first.

```
(select * from t1 union all select * from t1) intersect select * from t2;
+---+
| a |
+---+
| 1 |
+---+
1 rows in set (0.00 sec)
```

11.5.4.17.5 Use Order By and Limit

TiDB supports using `ORDER BY` or `LIMIT` clause in set operations. These two clauses must be at the end of the entire statement.

```
(select * from t1 union all select * from t1 intersect select * from t2)
    ↪ order by a limit 2;
+---+
| a |
+---+
| 1 |
| 1 |
+---+
2 rows in set (0.00 sec)
```

11.5.4.18 List of Expressions for Pushdown

When TiDB reads data from TiKV, TiDB tries to push down some expressions (including calculations of functions or operators) to be processed to TiKV. This reduces the amount of transferred data and offloads processing from a single TiDB node. This document introduces the expressions that TiDB already supports pushing down and how to prohibit specific expressions from being pushed down using blocklist.

11.5.4.18.1 Supported expressions for pushdown

Expression Type	Operations
Logical operators	AND (&&), OR (), NOT (!)
Comparison functions and operators	<, <=, =, != (<>), >, >=, <=>, IN(), IS NULL, LIKE, IS TRUE, IS FALSE, COALESCE()
Numeric functions and operators	+, -, *, /, ABS(), CEIL(), CEILING(), FLOOR()
Control flow functions	CASE, IF(), IFNULL()
JSON functions	JSON_TYPE(json_val), JSON_EXTRACT(json_doc, path[, path] ...), JSON_OBJECT(key, val[, key, val] ...), JSON_ARRAY([val[, val] ...]), JSON_MERGE(json_doc, json_doc[, json_doc] ...), JSON_SET(json_doc, path, val[, path, val] ...), JSON_INSERT(json_doc, path, val[, path, val] ...), JSON_REPLACE(json_doc, path, val[, path, val] ...), JSON_REMOVE(json_doc, path[, path] ...)
Date and time functions	DATE_FORMAT()

11.5.4.18.2 Blocklist specific expressions

If unexpected behavior occurs during the calculation of a function caused by its pushdown, you can quickly restore the application by blocklisting that function. Specifically, you can prohibit an expression from being pushed down by adding the corresponding functions or operator to the blocklist `mysql.expr_pushdown_blacklist`.

The schema of `mysql.expr_pushdown_blacklist` is as follows:

```
tidb> desc mysql.expr_pushdown_blacklist;
+-----+-----+-----+-----+-----+
| Field | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| name  | char(100) | NO   |     | NULL    |        |
| store_type | char(100) | NO   |     |          | tikv,tiflash,tidb |
| reason | varchar(200) | YES  |     | NULL    |        |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

Field description:

- **name**: the name of the function that is prohibited from being pushed down.
- **store_type**: specifies to which storage engine the function is prohibited from being pushed down. Currently, TiDB supports the three storage engines: `tikv`, `tidb`, and `tiflash`. `store_type` is case-insensitive. If a function is prohibited from being pushed down to multiple storage engines, use a comma to separate each engine.
- **reason** : The reason why the function is blocklisted.

Add to the blocklist

To add one or more functions or operators to the blocklist, perform the following steps:

1. Insert the function or operator name and the collection of storage types to be prohibited from the function pushdown to `mysql.expr_pushdown_blacklist`.
2. Execute the `admin reload expr_pushdown_blacklist;` command.

Remove from the blocklist

To remove one or more functions or operators from the blocklist, perform the following steps:

1. Delete the function or operator name in `mysql.expr_pushdown_blacklist`.
2. Execute the `admin reload expr_pushdown_blacklist;` command.

Blocklist usage examples

The following example demonstrates how to add the `<` and `>` operators to the blocklist, then remove `>` from the blocklist.

You can see whether the blocklist takes effect by checking the results returned by `EXPLAIN` statement (See [Understanding EXPLAIN results](#)).

```

tidb> create table t(a int);
Query OK, 0 rows affected (0.06 sec)

tidb> explain select * from t where a < 2 and a > 2;
+--+
| id           | estRows | task      | access object | operator info
|             |          |          |              |
+--+
| TableReader_7    | 0.00    | root      |               | data:Selection_6
|             |          |          |              |
| -Selection_6     | 0.00    | cop[tikv] |               | gt(ssb_1.t.a, 2)
|   , lt(ssb_1.t.a, 2) |
| -TableFullScan_5 | 10000.00 | cop[tikv] | table:t    | keep order:false
|   , stats:pseudo |
+--+
|             |          |          |              |
|             |          |          |
3 rows in set (0.00 sec)

tidb> insert into mysql.expr_pushdown_blacklist values('<', 'tikv', ''), ('>',
|   , 'tikv', '');
Query OK, 2 rows affected (0.01 sec)
Records: 2 Duplicates: 0 Warnings: 0

tidb> admin reload expr_pushdown_blacklist;
Query OK, 0 rows affected (0.00 sec)

tidb> explain select * from t where a < 2 and a > 2;
+--+
| id           | estRows | task      | access object | operator info
|             |          |          |              |
+--+
| Selection_7     | 10000.00 | root      |               | gt(ssb_1.t.a, 2),
|   , lt(ssb_1.t.a, 2) |
| -TableReader_6    | 10000.00 | root      |               | data:
|   TableFullScan_5 |
+--+
|             |          |          |              |
|             |          |          |

```

```

|   -TableFullScan_5  | 10000.00 | cop[tikv] | table:t | keep order:false
|   , stats:pseudo |
+--+
|   +-----+-----+-----+
|   |
|   3 rows in set (0.00 sec)

tidb> delete from mysql.expr_pushdown_blacklist where name = '>';
Query OK, 1 row affected (0.01 sec)

tidb> admin reload expr_pushdown_blacklist;
Query OK, 0 rows affected (0.00 sec)

tidb> explain select * from t where a < 2 and a > 2;
+--+
|   +-----+-----+-----+-----+
|   |
|   | id           | estRows | task      | access object | operator info
|   |             |          |           |              | 
+--+
|   +-----+-----+-----+-----+
|   |
|   | Selection_8    | 0.00    | root      |                  | lt(ssb_1.t.a,
|   |   2)          |          |           |                  | 
|   | -TableReader_7 | 0.00    | root      |                  | data:
|   |   Selection_6  |          |           |                  | 
|   |   -Selection_6 | 0.00    | cop[tikv] |                  | gt(ssb_1.t.a,
|   |   2)          |          |           |                  | 
|   |   -TableFullScan_5 | 10000.00 | cop[tikv] | table:t | keep order:
|   |   , stats:pseudo |
+--+
|   +-----+-----+-----+
|   |
|   4 rows in set (0.00 sec)

```

Note:

- `admin reload expr_pushdown_blacklist` only takes effect on the TiDB server that executes this SQL statement. To make it apply to all TiDB servers, execute the SQL statement on each TiDB server.
- The feature of blocklisting specific expressions is supported in TiDB 3.0.0 or later versions.

- TiDB 3.0.3 or earlier versions does not support adding some of the operators (such as “>”, “+”, “is null”) to the blocklist by using their original names. You need to use their aliases (case-sensitive) instead, as shown in the following table:

Operator Name	Aliases
<	lt
>	gt
<=	le
>=	ge
=	eq
!=	ne
<>	ne
<=>	nulleq
	bitor
&&	bitand
	or
!	not
in	in
+	plus
-	minus
	mul
/	div
DIV	intdiv
IS NULL	isnull
IS TRUE	istrue
IS FALSE	isfalse

11.5.4.19 TiDB Specific Functions

The following functions are TiDB extensions, and are not present in MySQL:

Function name	Function description
TIDB_BOUNDED_STALENESS	TIDB_BOUNDED_STALENESS function → () instructs TiDB to read the data as new as possible within the time range. See also: Read Historical Data Using the AS OF TIMESTAMP Clause

Function name	Function description
TIDB_DECODE_KEY → (str)	The TIDB_DECODE_KEY function can be used to decode a TiDB-encoded key entry into a JSON structure containing <code>_tidb_rowid</code> and <code>table_id</code> . These encoded keys can be found in some system tables and in logging outputs.
TIDB_DECODE_PLAN → (str)	The TIDB_DECODE_PLAN function can be used to decode a TiDB execution plan.
TIDB_IS_DDL_OWNER → ()	The TIDB_IS_DDL_OWNER function can be used to check whether or not the TiDB instance you are connected to is the one that is the DDL Owner. The DDL Owner is the TiDB instance that is tasked with executing DDL statements on behalf of all other nodes in the cluster.
TIDB_PARSE_TSO → (num)	The TIDB_PARSE_TSO function can be used to extract the physical timestamp from a TiDB TSO timestamp. See also: tidb_current_ts .
TIDB_VERSION()	The TIDB_VERSION function returns the TiDB version with additional build information.
TIDB_DECODE_SQL_DIGESTS → (digests, → stmtTruncateLen →)	The TIDB_DECODE_SQL_DIGESTS() function is used to query the normalized SQL statements (a form without formats and arguments) corresponding to the set of SQL digests in the cluster.
VITNESS_HASH (→ str)	The VITNESS_HASH function returns the hash of a string that is compatible with Vitess' HASH function. This is intended to help the data migration from Vitess.

11.5.4.19.1 Examples

This section provides examples for some of the functions above.

TIDB_DECODE_KEY

In the following example, the table `t1` has a hidden `rowid` that is generated by TiDB. The TIDB_DECODE_KEY is used in the statement. From the result, you can see that the hidden `rowid` is decoded and output, which is a typical result for the non-clustered primary key.

```
SELECT START_KEY, TIDB_DECODE_KEY(START_KEY) FROM information_schema.  
→ tikv_region_status WHERE table_name='t1' AND REGION_ID=2\G
```

```
***** 1. row *****
START_KEY: 7480000000000000
    ↪ FF3B5F72800000000FF1DE3F10000000000FA
TIDB_DECODE_KEY(START_KEY): {"_tidb_rowid":1958897,"table_id":"59"}
1 row in set (0.00 sec)
```

In the following example, the table t2 has a compound clustered primary key. From the JSON output, you can see a handle that contains the name and value for both of the columns that are part of the primary key.

```
show create table t2\G
```

```
***** 1. row *****
Table: t2
Create Table: CREATE TABLE `t2` (
  `id` binary(36) NOT NULL,
  `a` tinyint(3) unsigned NOT NULL,
  `v` varchar(512) DEFAULT NULL,
  PRIMARY KEY (`a`,`id`) /*T![clustered_index] CLUSTERED */
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin
1 row in set (0.001 sec)
```

```
select * from information_schema.tikv_region_status where table_name='t2'
    ↪ limit 1\G
```

```
***** 1. row *****
REGION_ID: 48
START_KEY: 7480000000000000
    ↪ FF3E5F72040000000FF0000000601633430FF3338646232FF2D64FF3531632D3131
    ↪
END_KEY:
TABLE_ID: 62
DB_NAME: test
TABLE_NAME: t2
IS_INDEX: 0
INDEX_ID: NULL
INDEX_NAME: NULL
EPOCH_CONF_VER: 1
EPOCH_VERSION: 38
WRITTEN_BYTES: 0
READ_BYTES: 0
APPROXIMATE_SIZE: 136
APPROXIMATE_KEYS: 479905
REPLICATIONSTATUS_STATE: NULL
```

```
REPLICATIONSTATUS_STATEID: NULL
1 row in set (0.005 sec)
```

```
select tidb_decode_key('7480000000000000
    ↪ FF3E5F72040000000FF000000601633430FF3338646232FF2D64FF3531632D3131FF65FF622D3863
    ↪ ');
```

```
+--+
|   ↪
|   ↪
| tidb_decode_key('7480000000000000
|     ↪ FF3E5F72040000000FF000000601633430FF3338646232FF2D64FF3531632D3131FF65FF622D3863
|     ↪ ') |
+--+
|   ↪
|   ↪
| { "handle": {"a": "6", "id": "c4038db2-d51c-11eb-8c75-80e65018a9be"}, "table_id"
|     ↪ :62}
|     ↪
|     ↪ |
+--+
|   ↪
|   ↪
1 row in set (0.001 sec)
```

TIDB_DECODE_PLAN

You can find TiDB execution plans in encoded form in the slow query log. The `TIDB_DECODE_PLAN()` function is then used to decode the encoded plans into a human-readable form.

This function is useful because a plan is captured at the time the statement is executed. Re-executing the statement in `EXPLAIN` might produce different results as data distribution and statistics evolves over time.

```
SELECT tidb_decode_plan('8
    ↪ QIYMAkzMV83CQEH8E85LjAOCWRhdGE6U2VsZWN0aW9uXzYJOTYwCXRpBU6NzEzLjHCtXMsIGxvb3Bz0j
    ↪ ') \G
```

```
***** 1. row *****
tidb_decode_plan('8
    ↪ QIYMAkzMV83CQEH8E85LjAOCWRhdGE6U2VsZWN0aW9uXzYJOTYwCXRpBU6NzEzLjHCtXMsIGxvb3Bz0j
    ↪ : id task estRows operator info                                actRows
    ↪ execution info
    ↪
    ↪ memory      disk
```

```

TableReader_7          root      319.04  data:Selection_6
    ↳                      960      time:713.1µs, loops:2, cop_task: {
        ↳ num: 1, max: 568.5µs, proc_keys: 0, rpc_num: 1, rpc_time: 549.1µs,
        ↳ copr_cache_hit_ratio: 0.00} 3.99 KB N/A
- Selection_6          cop[tikv]  319.04  lt(test.t.a, 10000)
    ↳                      960      tikv_task:{time:313.8µs, loops:960}
    ↳
    ↳ N/A      N/A
- TableFullScan_5      cop[tikv]  960      table:t, keep_order:false,
    ↳ stats:pseudo 960      tikv_task:{time:153µs, loops:960}
    ↳
    ↳ N/A      N/A

```

TIDB_PARSE_TSO

The `TIDB_PARSE_TSO` function can be used to extract the physical timestamp from a TiDB TSO timestamp. TSO stands for Time Stamp Oracle and is a monotonically increasing timestamp given out by PD (Placement Driver) for every transaction.

A TSO is a number that consists of two parts:

- A physical timestamp
- A logical counter

```
BEGIN;
SELECT TIDB_PARSE_TSO(@@tidb_current_ts);
ROLLBACK;
```

```
+-----+
| TIDB_PARSE_TSO(@@tidb_current_ts) |
+-----+
| 2021-05-26 11:33:37.776000       |
+-----+
1 row in set (0.0012 sec)
```

Here `TIDB_PARSE_TSO` is used to extract the physical timestamp from the timestamp number that is available in the `tidb_current_ts` session variable. Because timestamps are given out per transaction, this function is running in a transaction.

TIDB_VERSION

The `TIDB_VERSION` function can be used to get the version and build details of the TiDB server that you are connected to. You can use this function when reporting issues on GitHub.

```
SELECT TIDB_VERSION()\G
```

```
***** 1. row *****
TIDB_VERSION(): Release Version: v5.1.0-alpha-13-gd5e0ed0aa-dirty
Edition: Community
Git Commit Hash: d5e0ed0aaed72d2f2dfe24e9deec31cb6cb5fdf0
Git Branch: master
UTC Build Time: 2021-05-24 14:39:20
GoVersion: go1.13
Race Enabled: false
TiKV Min Version: v3.0.0-60965b006877ca7234adaced7890d7b029ed1306
Check Table Before Drop: false
1 row in set (0.00 sec)
```

TIDB_DECODE_SQL_DIGESTS

The `TIDB_DECODE_SQL_DIGESTS()` function is used to query the normalized SQL statements (a form without formats and arguments) corresponding to the set of SQL digests in the cluster. This function accepts 1 or 2 arguments:

- `digests`: A string. This parameter is in the format of a JSON string array, and each string in the array is a SQL digest.
- `stmtTruncateLength`: An integer (optional). It is used to limit the length of each SQL statement in the returned result. If a SQL statement exceeds the specified length, the statement is truncated. 0 means that the length is unlimited.

This function returns a string, which is in the format of a JSON string array. The i -th item in the array is the normalized SQL statement corresponding to the i -th element in the `digests` parameter. If an element in the `digests` parameter is not a valid SQL digest or the system cannot find the corresponding SQL statement, the corresponding item in the returned result is `null`. If the truncation length is specified ($\text{stmtTruncateLength} > 0$), for each statement in the returned result that exceeds this length, the first `stmtTruncateLength` characters are retained and the suffix "..." is added at the end to indicate the truncation. If the `digests` parameter is `NULL`, the returned value of the function is `NULL`.

Note:

- Only users with the `PROCESS` privilege can use this function.
- When `TIDB_DECODE_SQL_DIGESTS` is executed, TiDB queries the statement corresponding to each SQL digest from the statement summary tables, so there is no guarantee that the corresponding statement can always be found for any SQL digest. Only the statements that have been executed in the cluster can be found, and whether these SQL statements can be queried or not is also affected by the related configuration of the

statement summary tables. For the detailed description of the statement summary table, see [Statement Summary Tables](#).

- This function has a high overhead. In queries with a large number of rows (for example, querying the full table of `information_schema.cluster_tidb_trx` on a large and busy cluster), using this function might cause the queries to run for too long. Use it with caution.
 - This function has a high overhead because every time it is called, it internally queries the `STATEMENTS_SUMMARY`, `STATEMENTS_SUMMARY_HISTORY`, `CLUSTER_STATEMENTS_SUMMARY`, and `CLUSTER_STATEMENTS_SUMMARY_HISTORY` tables, and the query involves the `UNION` operation. This function currently does not support vectorization, that is, when calling this function for multiple rows of data, the above query is performed separately for each row.

```
set @digests = '['
→ e6f07d43b5c21db0fbb9a31feac2dc599787763393dd5acbfad80e247eb02ad5", "38
→ b03afa5debbdf0326a014dbe5012a62c51957f1982b3093e748460f8b00821",
→ e5796985ccafe2f71126ed6c0ac939ffa015a8c0744a24b7aee6d587103fd2f7"]';

select tidb_decode_sql_digests(@digests);
```

```
+-----+
| tidb_decode_sql_digests(@digests) |
+-----+
| [{"begin":null,"select * from `t`"}] |
+-----+
1 row in set (0.00 sec)
```

In the above example, the parameter is a JSON array containing 3 SQL digests, and the corresponding SQL statements are the three items in the query results. But the SQL statement corresponding to the second SQL digest cannot be found from the cluster, so the second item in the result is `null`.

```
select tidb_decode_sql_digests(@digests, 10);
```

```
+-----+
| tidb_decode_sql_digests(@digests, 10) |
+-----+
| [{"begin":null,"select * f..."}] |
+-----+
1 row in set (0.01 sec)
```

The above call specifies the second parameter (that is, the truncation length) as 10, and the length of the third statement in the query result is greater than 10. Therefore, only the first 10 characters are retained, and "..." is added at the end, which indicates the truncation.

See also:

- [Statement Summary Tables](#)
- [INFORMATION_SCHEMA.TIDB_TRX](#)

11.5.5 Clustered Indexes

TiDB supports the clustered index feature since v5.0. This feature controls how data is stored in tables containing primary keys. It provides TiDB the ability to organize tables in a way that can improve the performance of certain queries.

The term *clustered* in this context refers to the *organization of how data is stored* and not *a group of database servers working together*. Some database management systems refer to clustered indexes as *index-organized tables* (IOT).

Currently, tables containing primary keys in TiDB are divided into the following two categories:

- **NONCLUSTERED:** The primary key of the table is non-clustered index. In tables with non-clustered indexes, the keys for row data consist of internal `_tidb_rowid` implicitly assigned by TiDB. Because primary keys are essentially unique indexes, tables with non-clustered indexes need at least two key-value pairs to store a row, which are:
 - `_tidb_rowid` (key) - row data (value)
 - Primary key data (key) - `_tidb_rowid` (value)
- **CLUSTERED:** The primary key of the table is clustered index. In tables with clustered indexes, the keys for row data consist of primary key data given by the user. Therefore, tables with clustered indexes need only one key-value pair to store a row, which is:
 - Primary key data (key) - row data (value)

Note:

TiDB supports clustering only by a table's PRIMARY KEY. With clustered indexes enabled, the terms *the PRIMARY KEY* and *the clustered index* might be used interchangeably. PRIMARY KEY refers to the constraint (a logical property), and clustered index describes the physical implementation of how the data is stored.

11.5.5.1 User scenarios

Compared to tables with non-clustered indexes, tables with clustered indexes offer greater performance and throughput advantages in the following scenarios:

- When data is inserted, the clustered index reduces one write of the index data from the network.
- When a query with an equivalent condition only involves the primary key, the clustered index reduces one read of index data from the network.
- When a query with a range condition only involves the primary key, the clustered index reduces multiple reads of index data from the network.
- When a query with an equivalent or range condition only involves the primary key prefix, the clustered index reduces multiple reads of index data from the network.

On the other hand, tables with clustered indexes have certain disadvantages. See the following:

- There might be write hotspot issues when inserting a large number of primary keys with close values.
- The table data takes up more storage space if the data type of the primary key is larger than 64 bits, especially when there are multiple secondary indexes.

11.5.5.2 Usages

11.5.5.3 Create a table with clustered indexes

Since TiDB v5.0, you can add non-reserved keywords CLUSTERED or NONCLUSTERED after PRIMARY KEY in a CREATE TABLE statement to specify whether the table's primary key is a clustered index. For example:

```
CREATE TABLE t (a BIGINT PRIMARY KEY CLUSTERED, b VARCHAR(255));
CREATE TABLE t (a BIGINT PRIMARY KEY NONCLUSTERED, b VARCHAR(255));
CREATE TABLE t (a BIGINT KEY CLUSTERED, b VARCHAR(255));
CREATE TABLE t (a BIGINT KEY NONCLUSTERED, b VARCHAR(255));
CREATE TABLE t (a BIGINT, b VARCHAR(255), PRIMARY KEY(a, b) CLUSTERED);
CREATE TABLE t (a BIGINT, b VARCHAR(255), PRIMARY KEY(a, b) NONCLUSTERED);
```

Note that keywords KEY and PRIMARY KEY have the same meaning in the column definition.

You can also use the [comment syntax](#) in TiDB to specify the type of the primary key. For example:

```
CREATE TABLE t (a BIGINT PRIMARY KEY /*T! [clustered_index] CLUSTERED */, b
    → VARCHAR(255));
CREATE TABLE t (a BIGINT PRIMARY KEY /*T! [clustered_index] NONCLUSTERED */
    → , b VARCHAR(255));
```

```
CREATE TABLE t (a BIGINT, b VARCHAR(255), PRIMARY KEY(a, b) /*T! [
    ↵ clustered_index] CLUSTERED */;
CREATE TABLE t (a BIGINT, b VARCHAR(255), PRIMARY KEY(a, b) /*T! [
    ↵ clustered_index] NONCLUSTERED */;
```

For statements that do not explicitly specify the keyword CLUSTERED/NONCLUSTERED, the default behavior is controlled by the system variable `@@global.tidb_enable_clustered_index`. Supported values for this variable are as follows:

- OFF indicates that primary keys are created as non-clustered indexes by default.
- ON indicates that primary keys are created as clustered indexes by default.
- INT_ONLY indicates that the behavior is controlled by the configuration item `alter-primary-key`. If `alter-primary-key` is set to `true`, primary keys are created as non-clustered indexes by default. If it is set to `false`, only the primary keys which consist of an integer column are created as clustered indexes.

The default value of `@@global.tidb_enable_clustered_index` is INT_ONLY.

11.5.5.3.1 Add or drop clustered indexes

TiDB does not support adding or dropping clustered indexes after tables are created. Nor does it support the mutual conversion between clustered indexes and non-clustered indexes. For example:

```
ALTER TABLE t ADD PRIMARY KEY(b, a) CLUSTERED; -- Currently not supported.
ALTER TABLE t DROP PRIMARY KEY; -- If the primary key is a clustered index
    ↵ , then not supported.
ALTER TABLE t DROP INDEX `PRIMARY`; -- If the primary key is a clustered
    ↵ index, then not supported.
```

11.5.5.3.2 Add or drop non-clustered indexes

TiDB supports adding or dropping non-clustered indexes after tables are created. You can explicitly specify the keyword NONCLUSTERED or omit it. For example:

```
ALTER TABLE t ADD PRIMARY KEY(b, a) NONCLUSTERED;
ALTER TABLE t ADD PRIMARY KEY(b, a); -- If you omit the keyword, the
    ↵ primary key is a non-clustered index by default.
ALTER TABLE t DROP PRIMARY KEY;
ALTER TABLE t DROP INDEX `PRIMARY`;
```

11.5.5.3.3 Check whether the primary key is a clustered index

You can check whether the primary key of a table is a clustered index using one of the following methods:

- Execute the command SHOW CREATE TABLE.
- Execute the command SHOW INDEX FROM.
- Query the TIDB_PK_TYPE column in the system table information_schema.tables.

By running the command SHOW CREATE TABLE, you can see whether the attribute of PRIMARY KEY is CLUSTERED or NONCLUSTERED. For example:

```
mysql> SHOW CREATE TABLE t;
+----+
| Table | Create Table
+----+
| t    | CREATE TABLE `t` (
  `a` bigint(20) NOT NULL,
  `b` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`a`) /*T![clustered_index] CLUSTERED */
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_bin |
+----+
1 row in set (0.01 sec)
```

By running the command SHOW INDEX FROM, you can check whether the result in the column Clustered shows YES or NO. For example:

```
mysql> SHOW INDEX FROM t;
+----+
| Table | Non_unique | Key_name | Seq_in_index | Column_name | Collation | |
|       | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
|       | Index_comment | Visible | Expression | Clustered |
+----+
| t    |          0 | PRIMARY |           1 | a          | A          |          0
|       |    NULL |    NULL |        | BTREE      |           | YES
|       |    NULL |     YES |        |           |           |           |
+----+
```

```
1 row in set (0.01 sec)
```

You can also query the column TIDB_PK_TYPE in the system table `information_schema.tables` to see whether the result is CLUSTERED or NONCLUSTERED. For example:

```
mysql> SELECT TIDB_PK_TYPE FROM information_schema.tables WHERE
    ↪ table_schema = 'test' AND table_name = 't';
+-----+
| TIDB_PK_TYPE |
+-----+
| CLUSTERED   |
+-----+
1 row in set (0.03 sec)
```

11.5.5.4 Limitations

Currently, there are several different types of limitations for the clustered index feature. See the following:

- Situations that are not supported and not in the support plan:
 - Using clustered indexes together with the attribute `SHARD_ROW_ID_BITS` is not supported. Also, the attribute `PRE_SPLIT_REGIONS` does not take effect on tables with clustered indexes.
 - Downgrading tables with clustered indexes is not supported. If you need to downgrade such tables, use logical backup tools to migrate data instead.
- Situations that are not supported yet but in the support plan:
 - Adding, dropping, and altering clustered indexes using `ALTER TABLE` statements are not supported.
- Limitations for specific versions:
 - In v5.0, using the clustered index feature together with TiDB Binlog is not supported. After TiDB Binlog is enabled, TiDB only allows creating a single integer column as the clustered index of a primary key. TiDB Binlog does not replicate data changes (such as insertion, deletion, and update) on existing tables with clustered indexes to the downstream. If you need to replicate tables with clustered indexes to the downstream, upgrade your cluster to v5.1 or use `TiCDC` for replication instead.

After TiDB Binlog is enabled, if the clustered index you create is not a single integer primary key, TiDB returns the following error:

```
mysql> CREATE TABLE t (a VARCHAR(255) PRIMARY KEY CLUSTERED);
ERROR 8200 (HY000): Cannot create clustered index table when the binlog is
    ↪ ON
```

If you use clustered indexes together with the attribute `SHARD_ROW_ID_BITS`, TiDB reports the following error:

```
mysql> CREATE TABLE t (a VARCHAR(255) PRIMARY KEY CLUSTERED
    ↵ SHARD_ROW_ID_BITS = 3;
ERROR 8200 (HY000): Unsupported shard_row_id_bits for table with primary
    ↵ key as row id
```

11.5.5.5 Compatibility

11.5.5.5.1 Compatibility with earlier and later TiDB versions

TiDB supports upgrading tables with clustered indexes but not downgrading such tables, which means that data in tables with clustered indexes on a later TiDB version is not available on an earlier one.

The clustered index feature is partially supported in TiDB v3.0 and v4.0. It is enabled by default when the following requirements are fully met:

- The table contains a `PRIMARY KEY`.
- The `PRIMARY KEY` consists of only one column.
- The `PRIMARY KEY` is an `INTEGER`.

Since TiDB v5.0, the clustered index feature is fully supported for all types of primary keys, but the default behavior is consistent with TiDB v3.0 and v4.0. To change the default behavior, you can configure the system variable `@@tidb_enable_clustered_index` to `ON` or `OFF`. For more details, see [Create a table with clustered indexes](#).

11.5.5.5.2 Compatibility with MySQL

TiDB specific comment syntax supports wrapping the keywords `CLUSTERED` and `NONCLUSTERED` in a comment. The result of `SHOW CREATE TABLE` also contains TiDB specific SQL comments. MySQL databases and TiDB databases of an earlier version will ignore these comments.

11.5.5.5.3 Compatibility with TiDB ecosystem tools

The clustered index feature is only compatible with the following ecosystem tools in v5.0 and later versions:

- Backup and restore tools: BR, Dumpling, and TiDB Lightning.
- Data migration and replication tools: DM and TiCDC.

However, you cannot convert a table with non-clustered indexes to a table with clustered indexes by backing up and restoring the table using the v5.0 BR tool, and vice versa.

11.5.5.4 Compatibility with other TiDB features

For a table with a combined primary key or a single non-integer primary key, if you change the primary key from a non-clustered index to a clustered index, the keys of its row data change as well. Therefore, `SPLIT TABLE BY/BETWEEN` statements that are executable in TiDB versions earlier than v5.0 are no longer workable in v5.0 and later versions of TiDB. If you want to split a table with clustered indexes using `SPLIT TABLE BY/BETWEEN`, you need to provide the value of the primary key column, instead of specifying an integer value. See the following example:

```
mysql> create table t (a int, b varchar(255), primary key(a, b) clustered);
Query OK, 0 rows affected (0.01 sec)
mysql> split table t between (0) and (1000000) regions 5;
ERROR 1105 (HY000): Split table region lower value count should be 2
mysql> split table t by (0), (50000), (100000);
ERROR 1136 (21S01): Column count doesn't match value count at row 0
mysql> split table t between (0, 'aaa') and (1000000, 'zzz') regions 5;
+-----+-----+
| TOTAL_SPLIT_REGION | SCATTER_FINISH_RATIO |
+-----+-----+
|           4 |           1 |
+-----+-----+
1 row in set (0.00 sec)
mysql> split table t by (0, ''), (50000, ''), (100000, '');
+-----+-----+
| TOTAL_SPLIT_REGION | SCATTER_FINISH_RATIO |
+-----+-----+
|           3 |           1 |
+-----+-----+
1 row in set (0.01 sec)
```

The attribute `AUTO_RANDOM` can only be used on clustered indexes. Otherwise, TiDB returns the following error:

```
mysql> create table t (a bigint primary key nonclustered auto_random);
ERROR 8216 (HY000): Invalid auto random: column a is not the integer
→ primary key, or the primary key is nonclustered
```

11.5.6 Constraints

TiDB supports almost the same constraint as MySQL.

11.5.6.1 NOT NULL

NOT NULL constraints supported by TiDB are the same as those supported by MySQL.

For example:

```
CREATE TABLE users (
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    age INT NOT NULL,
    last_login TIMESTAMP
);
```

```
INSERT INTO users (id,age,last_login) VALUES (NULL,123,NOW());
```

Query OK, 1 row affected (0.02 sec)

```
INSERT INTO users (id,age,last_login) VALUES (NULL,NULL,NOW());
```

ERROR 1048 (23000): Column 'age' cannot be null

```
INSERT INTO users (id,age,last_login) VALUES (NULL,123,NULL);
```

Query OK, 1 row affected (0.03 sec)

- The first `INSERT` statement succeeds because it is possible to assign `NULL` to the `AUTO_INCREMENT` column. TiDB generates sequence numbers automatically.
- The second `INSERT` statement fails because the `age` column is defined as `NOT NULL`.
- The third `INSERT` statement succeeds because the `last_login` column is not explicitly defined as `NOT NULL`. `NULL` values are allowed by default.

11.5.6.2 CHECK

TiDB parses but ignores `CHECK` constraints. This is MySQL 5.7 compatible behavior.

For example:

```
DROP TABLE IF EXISTS users;
CREATE TABLE users (
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(60) NOT NULL,
    UNIQUE KEY (username),
    CONSTRAINT min_username_length CHECK (CHARACTER_LENGTH(username) >=4)
);
INSERT INTO users (username) VALUES ('a');
SELECT * FROM users;
```

11.5.6.3 UNIQUE KEY

Depending on the transaction mode and the value of `tidb_constraint_check_in_place` → , TiDB might check UNIQUE constraints **lazily**. This helps improve performance by batching network access.

For example:

```
DROP TABLE IF EXISTS users;
CREATE TABLE users (
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(60) NOT NULL,
    UNIQUE KEY (username)
);
INSERT INTO users (username) VALUES ('dave'), ('sarah'), ('bill');
```

With the default of pessimistic locking:

```
BEGIN;
INSERT INTO users (username) VALUES ('jane'), ('chris'), ('bill');
```

```
ERROR 1062 (23000): Duplicate entry 'bill' for key 'username'
```

With optimistic locking and `tidb_constraint_check_in_place=0`:

```
BEGIN OPTIMISTIC;
INSERT INTO users (username) VALUES ('jane'), ('chris'), ('bill');
```

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

```
Query OK, 3 rows affected (0.00 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

```
INSERT INTO users (username) VALUES ('steve'),('elizabeth');
```

```
Query OK, 2 rows affected (0.00 sec)
Records: 2 Duplicates: 0 Warnings: 0
```

```
COMMIT;
```

```
ERROR 1062 (23000): Duplicate entry 'bill' for key 'username'
```

In the optimistic example, the unique check was delayed until the transaction is committed. This resulted in a duplicate key error, because the value `bill` was already present.

You can disable this behavior by setting `tidb_constraint_check_in_place` to 1. This variable setting does not take effect on pessimistic transactions, because in the pessimistic

transaction mode the constraints are always checked when the statement is executed. When `tidb_constraint_check_in_place=1`, the unique constraint is checked when the statement is executed.

For example:

```
DROP TABLE IF EXISTS users;
CREATE TABLE users (
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(60) NOT NULL,
    UNIQUE KEY (username)
);
INSERT INTO users (username) VALUES ('dave'), ('sarah'), ('bill');
```

```
SET tidb_constraint_check_in_place = 1;
```

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

```
BEGIN OPTIMISTIC;
```

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

```
INSERT INTO users (username) VALUES ('jane'), ('chris'), ('bill');
```

```
ERROR 1062 (23000): Duplicate entry 'bill' for key 'username'
..
```

The first `INSERT` statement caused a duplicate key error. This causes additional network communication overhead and may reduce the throughput of insert operations.

11.5.6.4 PRIMARY KEY

Like MySQL, primary key constraints contain unique constraints, that is, creating a primary key constraint is equivalent to having a unique constraint. In addition, other primary key constraints of TiDB are also similar to those of MySQL.

For example:

```
CREATE TABLE t1 (a INT NOT NULL PRIMARY KEY);
```

```
Query OK, 0 rows affected (0.12 sec)
```

```
CREATE TABLE t2 (a INT NULL PRIMARY KEY);
```

```
ERROR 1171 (42000): All parts of a PRIMARY KEY must be NOT NULL; if you need
↪ NULL in a key, use UNIQUE instead
```

```
CREATE TABLE t3 (a INT NOT NULL PRIMARY KEY, b INT NOT NULL PRIMARY KEY);
```

ERROR 1068 (42000): Multiple primary key defined

```
CREATE TABLE t4 (a INT NOT NULL, b INT NOT NULL, PRIMARY KEY (a,b));
```

Query OK, 0 rows affected (0.10 sec)

- Table **t2** failed to be created, because column **a** is defined as the primary key and does not allow NULL values.
- Table **t3** failed to be created, because a table can only have one primary key.
- Table **t4** was created successfully, because even though there can be only one primary key, TiDB supports defining multiple columns as the composite primary key.

In addition to the rules above, TiDB currently only supports adding and deleting the primary keys of the **NONCLUSTERED** type. For example:

```
CREATE TABLE t5 (a INT NOT NULL, b INT NOT NULL, PRIMARY KEY (a,b)
    ↪ CLUSTERED);
ALTER TABLE t5 DROP PRIMARY KEY;
```

ERROR 8200 (HY000): Unsupported drop primary key when the table is using
 ↪ clustered index

```
CREATE TABLE t5 (a INT NOT NULL, b INT NOT NULL, PRIMARY KEY (a,b)
    ↪ NONCLUSTERED);
ALTER TABLE t5 DROP PRIMARY KEY;
```

Query OK, 0 rows affected (0.10 sec)

For more details about the primary key of the **CLUSTERED** type, refer to [clustered index](#).

11.5.6.5 FOREIGN KEY

Note:

TiDB has limited support for foreign key constraints.

TiDB supports creating **FOREIGN KEY** constraints in DDL commands.

For example:

```

CREATE TABLE users (
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    doc JSON
);
CREATE TABLE orders (
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    user_id INT NOT NULL,
    doc JSON,
    FOREIGN KEY fk_user_id (user_id) REFERENCES users(id)
);

```

```

SELECT table_name, column_name, constraint_name, referenced_table_name,
    ↪ referenced_column_name
FROM information_schema.key_column_usage WHERE table_name IN ('users', '
    ↪ orders');

```

↪	table_name	column_name	constraint_name	referenced_table_name	↪	referenced_column_name
↪	users	id	PRIMARY	NULL	↪	NULL
↪	orders	id	PRIMARY	NULL	↪	NULL
↪	orders	user_id	fk_user_id	users	↪	id

3 rows in set (0.00 sec)

TiDB also supports the syntax to DROP FOREIGN KEY and ADD FOREIGN KEY via the ALTER TABLE command.

```

ALTER TABLE orders DROP FOREIGN KEY fk_user_id;
ALTER TABLE orders ADD FOREIGN KEY fk_user_id (user_id) REFERENCES users(id
    ↪ );

```

11.5.6.5.1 Notes

- TiDB supports foreign keys to avoid errors caused by this syntax when you migrate data from other databases to TiDB.

However, TiDB does not perform constraint checking on foreign keys in DML statements. For example, even if there is no record with id=123 in the users table, the following transactions can be submitted successfully.

```
START TRANSACTION;
INSERT INTO orders (user_id, doc) VALUES (123, NULL);
COMMIT;
```

11.5.7 Generated Columns

Warning:

This is still an experimental feature. It is **NOT** recommended that you use it in the production environment.

This document introduces the concept and usage of generated columns.

11.5.7.1 Basic concepts

Unlike general columns, the value of the generated column is calculated by the expression in the column definition. When inserting or updating a generated column, you cannot assign a value, but only use `DEFAULT`.

There are two kinds of generated columns: virtual and stored. A virtual generated column occupies no storage and is computed when it is read. A stored generated column is computed when it is written (inserted or updated) and occupies storage. Compared with the virtual generated columns, the stored generated columns have better read performance, but take up more disk space.

You can create an index on a generated column whether it is virtual or stored.

11.5.7.2 Usage

One of the main usage of generated columns is to extract data from the JSON data type and indexing the data.

In both MySQL 5.7 and TiDB, columns of type JSON can not be indexed directly. That is, the following table schema is **not supported**:

```
CREATE TABLE person (
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    address_info JSON,
    KEY (address_info)
);
```

To index a JSON column, you must extract it as a generated column first.

Using the `city` field in `address_info` as an example, you can create a virtual generated column and add an index for it:

```
CREATE TABLE person (
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    address_info JSON,
    city VARCHAR(64) AS (JSON_UNQUOTE(JSON_EXTRACT(address_info, '$.city')))
    → ,
    KEY (city)
);
```

In this table, the `city` column is a **virtual generated column** and has an index. The following query can use the index to speed up the execution:

```
SELECT name, id FROM person WHERE city = 'Beijing';
```

```
EXPLAIN SELECT name, id FROM person WHERE city = 'Beijing';
```

→	id	estRows	task	access object
→	→	operator info		
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→	→	→	→	→
→				

```
CREATE TABLE person (
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    address_info JSON,
    city VARCHAR(64) AS (JSON_UNQUOTE(JSON_EXTRACT(address_info, '$.city')))
        → NOT NULL,
    KEY (city)
);
```

11.5.7.3 Validation of generated columns

Both INSERT and UPDATE statements check virtual column definitions. Rows that do not pass validation return errors:

```
mysql> INSERT INTO person (name, address_info) VALUES ('Morgan',
   → JSON_OBJECT('Country', 'Canada'));
ERROR 1048 (23000): Column 'city' cannot be null
```

11.5.7.4 Generated columns index replacement rule

When an expression in a query is equivalent to a generated column with an index, TiDB replaces the expression with the corresponding generated column, so that the optimizer can take that index into account during execution plan construction.

For example, the following example creates a generated column for the expression `a+1` and adds an index:

```
create table t(a int);
desc select a+1 from t where a+1=3;
+---+
| id          | estRows | task      | access object | operator info
|             |          |           |               |
+---+
| Projection_4 | 8000.00 | root      |               | plus(test.t.a,
|             |          |           |               |
| -TableReader_7 | 8000.00 | root      |               | data:
|             |          |           |               |
| -Selection_6 | 8000.00 | cop[tikv] |               | eq(plus(test.t
|             |          |           |               |
| -Selection_6 | 8000.00 | cop[tikv] |               | eq(plus(test.t
|             |          |           |               |
| -TableFullScan_5 | 10000.00 | cop[tikv] | table:t | keep order:
|             |          |           |               |
|             |          |           |               | false, stats:pseudo |
```

```
+--+
→ -----
→
4 rows in set (0.00 sec)

alter table t add column b bigint as (a+1) virtual;
alter table t add index idx_b(b);
desc select a+1 from t where a+1=3;
+--
→ -----
→
| id           | estRows | task      | access object      | operator
| info          |          |           |
+--
→ -----
→
| IndexReader_6   | 10.00  | root      |                     | index:
|   IndexRangeScan_5          |          |
| -IndexRangeScan_5  | 10.00  | cop[tikv] | table:t, index:idx_b(b) |
|   range:[3,3], keep order:false, stats:pseudo |
+--
→ -----
→
2 rows in set (0.01 sec)
```

Note:

Only when the expression type and the generated column type are strictly equal, the replacement is performed.

In the above example, the column type of `a` is int and the column type of `a+1` is bigint. If the type of the generated column is set to int, the replacement will not occur.

For type conversion rules, see [Type Conversion of Expression Evaluation](#).

11.5.7.5 Limitations

The current limitations of JSON and generated columns are as follows:

- You cannot add a stored generated column through `ALTER TABLE`.
- You can neither convert a stored generated column to a normal column through the `ALTER TABLE` statement nor convert a normal column to a stored generated column.

- You cannot modify the expression of a stored generated column through the `ALTER ↵ TABLE` statement.
- Not all **JSON functions** are supported;
- Currently, the generated column index replacement rule is valid only when the generated column is a virtual generated column. It is not valid on the stored generated column, but the index can still be used by directly using the generated column itself.

— title: SQL Mode summary: Learn SQL mode. —

11.5.8 SQL Mode

TiDB servers operate in different SQL modes and apply these modes differently for different clients. SQL mode defines the SQL syntaxes that TiDB supports and the type of data validation check to perform, as described below:

After TiDB is started, modify `SET [SESSION | GLOBAL] sql_mode='modes'` to set SQL mode.

Ensure that you have **SUPER** privilege when setting SQL mode at **GLOBAL** level, and your setting at this level only affects the connections established afterwards. Changes to SQL mode at **SESSION** level only affect the current client.

Modes are a series of different modes separated by commas (‘,’). You can use the `SELECT ↵ @@sql_mode` statement to check the current SQL mode. The default value of SQL mode: `ONLY_FULL_GROUP_BY, STRICT_TRANS_TABLES, NO_ZERO_IN_DATE, NO_ZERO_DATE, ↵ ERROR_FOR_DIVISION_BY_ZERO, NO_AUTO_CREATE_USER, NO_ENGINE_SUBSTITUTION`.

11.5.8.1 Important `sql_mode` values

- **ANSI**: This mode complies with standard SQL. In this mode, data is checked. If data does not comply with the defined type or length, the data type is adjusted or trimmed and a **warning** is returned.
- **STRICT_TRANS_TABLES**: Strict mode, where data is strictly checked. When any incorrect data is inserted into a table, an error is returned.
- **TRADITIONAL**: In this mode, TiDB behaves like a “traditional” SQL database system. An error instead of a warning is returned when any incorrect value is inserted into a column. Then, the `INSERT` or `UPDATE` statement is immediately stopped.

11.5.8.2 SQL mode table

Name	Description
PIPES_AS_CONCAT	
↪	" "
	as a
	string
	con-
	cate-
	na-
	tion
	oper-
	ator
(+)	(+)
(the	(the
same	same
as	as
CONCAT	CONCAT
↪ ()	↪ ()
↪),	↪),
not	not
as an	as an
OR	OR
(full	(full
sup-	sup-
port)	port)

Name	Description
<code>ANSI_QUOTES</code>	
<code>↪</code>	" as
	an
	iden-
	tifier.
	If
	<code>ANSI_QUOTES</code>
	<code>↪</code>
	is en-
	abled,
	only
	sin-
	gle
	quotes
	are
	treated
	as
	string
	liter-
	als,
	and
	dou-
	ble
	quotes
	are
	treated
	as
	iden-
	ti-
	fiers.
	There-
	fore,
	dou-
	ble
	quotes
	can-
	not
	be
	used
	to
	quote
	strings.
	(full
	sup-
	port)

Name	Description
IGNORE_L\$PACE	→ this mode is enabled, the system ignores space. For example: “user” and “user” are the same. (full support)

Name	Description
ONLY_FULL_GROUP_BY	
↪	non-aggregated
	col-
	umn
	that
	is re-
	ferred
	to in
	SELECT
	↪ ,
	HAVING
	↪ ,
	or
	ORDER
	↪
	↪ BY
	↪
	is ab-
	sent
	in
	GROUP
	↪
	↪ BY
	↪ ,
	this
	SQL
	state-
	ment
	is in-
	valid,
	be-
	cause
	it is
	ab-
	nor-
	mal
	for a
	col-
	umn
	to be
	ab-
	sent
	in
	GROUP
1869	
↪	
	↪ BY
	↪
	↪

Name	Description
NO_UNSIGNED_SUBTRACTION	
↪	not
	mark
	the
	re-
	sult
	as
	UNSIGNED
↪	
	if an
	operand
	has
	no
	sym-
	bol
	in
	sub-
	trac-
	tion.
	(full
	sup-
	port)

Name	Description
NO_DIR_INDEX_CREATE	<p>↪ all INDEX</p> <p>↪ DIRECTORY</p> <p>↪ and DATA</p> <p>↪ DIRECTORY</p> <p>↪ directives when a table is created.</p> <p>This option is only useful for secondary servers (syntax support only)</p>

Name	Description
NO_KEY_OPTIONS	
↪	you
	use
	the
	SHOW
↪	
↪	CREATE
↪	
↪	TABLE
↪	
	state-
	ment,
	MySQL-
	specific
	syn-
	taxes
	such
	as
	ENGINE
↪	
	are
	not
	ex-
	ported.
	Con-
	sider
	this
	op-
	tion
	when
	mi-
	grat-
	ing
	across
	DB
	types
	using
	mysql-
	dump.
	(syn-
	tax
	sup-
	port
	only)

Name	Description
NO_FIELD_OPTIONS	
↪	you
	use
	the
	SHOW
↪	
↪	CREATE
↪	
↪	TABLE
↪	
	state-
	ment,
	MySQL-
	specific
	syn-
	taxes
	such
	as
	ENGINE
↪	
	are
	not
	ex-
	ported.
	Con-
	sider
	this
	op-
	tion
	when
	mi-
	grat-
	ing
	across
	DB
	types
	using
	mysql-
	dump.
	(syn-
	tax
	sup-
	port
	only)

Name	Description
NO_TABLEOPTIONS	<p>→ you</p> <p>use</p> <p>the</p> <p>SHOW</p> <p>→</p> <p>→ CREATE</p> <p>→</p> <p>→ TABLE</p> <p>→</p> <p>state-</p> <p>ment,</p> <p>MySQL-</p> <p>specific</p> <p>syn-</p> <p>taxes</p> <p>such</p> <p>as</p> <p>ENGINE</p> <p>→</p> <p>are</p> <p>not</p> <p>ex-</p> <p>ported.</p> <p>Con-</p> <p>sider</p> <p>this</p> <p>op-</p> <p>tion</p> <p>when</p> <p>mi-</p> <p>grat-</p> <p>ing</p> <p>across</p> <p>DB</p> <p>types</p> <p>using</p> <p>mysql-</p> <p>dump.</p> <p>(syn-</p> <p>tax</p> <p>sup-</p> <p>port</p> <p>only)</p>

Name	Description
NO_AUTO_VALUE_ON_ZERO	<p>↪ this mode is enabled, when the value passed in the AUTO_INCREMENT</p> <p>↪ column is 0 or a specific value, the system directly writes this value to this column.</p> <p>When NULL is passed, the system automatically generates the next serial</p>

Name	Description
NO_BACKSLASH_ESCAPES	→ this mode is enabled, the \ backslash symbol only stands for itself. (full support)

Name	Description
STRICT_TRANS_TABLES	↪ the strict mode for the transaction storage engine and rolls back the entire state after an illegal value is inserted. (full support)

Name	Description
STRICT_FAILOVER_TABLES	→ trans- ac- tional ta- bles, rolls back the en- tire trans- ac- tion state- ment after an il- legal value is in- serted. (full sup- port)

Name	Description
------	-------------

NO_ZERO_DATE	← mode, where dates with a month or day part of 0 are not ac- cepted.
--------------	--

If
you
use
the
IGNORE
←
op-
tion,
TiDB
in-
serts
‘0000-
00-
00’
for a
simi-
lar
date.
In
non-
strict
mode,
this
date
is ac-
cepted
but a
warn-
ing is
re-

1879 turned.

(full
sup-

Name	Description
NO_ZERO_DATE	
↪	not use '0000-00-00' as a legal date in strict mode.
You can still insert a zero date with the IGNORE option.	In non-strict mode, this date is accepted but a warning is re-turned. (full support)

Name	Description
ALLOW_INVALID_DATES	→ this mode, the system does not check the validity of all dates.
	It only checks the month value ranging from 1 to 12 and the date value ranging from 1 to 31.
	The mode only applies to DATE and DATETIME → columns.
1881	All TIMESTAMP →

Name	Description
ERROR_DIVISION_BY_ZERO	
↪	this mode is enabled, the system returns an error when handling division by 0 in data-change operations (INSERT ↪ or UPDATE ↪). If this mode is not enabled, the system returns a warning and 1882 NULL is used.
1882	
NULL	
is	
used	
:	

Name	Description
NO_AUTO_CREATE_USER	↪ GRANT ↪ from auto- mati- cally creat- ing new users, ex- cept for the speci- fied pass- word (full sup- port)

Name	Description
HIGH_NOTPRECEDENCE	
↪	precedence of the NOT operator is such that expressions such as NOT
↪ a	
↪	↪ BETWEEN
↪	↪ b
↪	↪ AND
↪	↪ c
are	
parsed	
as	
NOT	
↪ (
↪ a	
↪	↪ BETWEEN
↪	↪ b
↪	↪ AND
↪	↪ c
↪).	

In some older versions of MySQL, this

Name	Description
NO_ENGINEREV_SUBSTITUTION	↪ the automatic replacement of storage engines if the required storage engine is disabled or not compiled. (syntax support only)

Name	Description
PAD_CHAR_TO_FULL_LENGTH	→ this mode is enabled, the system does not trim the trailing spaces for CHAR types. (syntax support only. This mode has been deprecated in MySQL 8.0.)

Name	Description
REAL_AS_FLOAT	
↳ REAL	
as	
the	
syn-	
onym	
of	
FLOAT	
↳ ,	
not	
the	
syn-	
onym	
of	
DOUBLE	
↳	
(full	
sup-	
port)	
POSTGRES_EQUIV	PostgreSQL equivalent
↳ to	
PIPES_AS_CONCAT	
↳ ,	
ANSI_QUOTES	
↳ ,	
IGNORE_SPACE	
↳ ,	
NO_KEY_OPTIONS	
↳ ,	
NO_TABLE_OPTIONS	
↳ ,	
NO_FIELD_OPTIONS	
↳	
(syn-	
tax	
sup-	
port	
only)	

Name	Description
MSSQL	Equivalent ↪ to PIPES_AS_CONCAT
	↪ , ANSI_QUOTES
	↪ , IGNORE_SPACE
	↪ , NO_KEY_OPTIONS
	↪ , NO_TABLE_OPTIONS
	↪ , NO_FIELD_OPTIONS
	↪ (syn- tax sup- port only)
DB2	Equivalent to PIPES_AS_CONCAT
	↪ , ANSI_QUOTES
	↪ , IGNORE_SPACE
	↪ , NO_KEY_OPTIONS
	↪ , NO_TABLE_OPTIONS
	↪ , NO_FIELD_OPTIONS
	↪ (syn- tax sup- port only)

Name	Description
MAXDB	Equivalent → to PIPES_AS_CONCAT → , ANSI_QUOTES → , IGNORE_SPACE → , NO_KEY_OPTIONS → , NO_TABLE_OPTIONS → , NO_FIELD_OPTIONS → , NO_AUTO_CREATE_USER → (full sup- port)
MySQL323	Equivalent → to NO_FIELD_OPTIONS → , HIGH_NOT_PRECEDENCE → (syn- tax sup- port only)
MySQL40	Equivalent → to NO_FIELD_OPTIONS → , HIGH_NOT_PRECEDENCE → (syn- tax sup- port only)

Name	Description
ANSI	Equivalent to REAL_AS_FLOAT ↪ , PIPES_AS_CONCAT ↪ , ANSI_QUOTES ↪ , IGNORE_SPACE ↪ (syn- tax sup- port only)
TRADITIONAL	Equivalent to STRICT_TRANS_TABLES ↪ , STRICT_ALL_TABLES ↪ , NO_ZERO_IN_DATE ↪ , NO_ZERO_DATE ↪ , ERROR_FOR_DIVISION_BY_ZERO ↪ , NO_AUTO_CREATE_USER ↪ (syn- tax sup- port only)

Name	Description
ORACLE Equivalent	
→	to
	PIPES_AS_CONCAT
→ ,	
	ANSI_QUOTES
→ ,	
	IGNORE_SPACE
→ ,	
	NO_KEY_OPTIONS
→ ,	
	NO_TABLE_OPTIONS
→ ,	
	NO_FIELD_OPTIONS
→ ,	
	NO_AUTO_CREATE_USER
→	
(syn-	
tax	
sup-	
port	
only)	

11.5.9 Table Attributes

The Table Attributes feature is introduced in TiDB v5.3.0. Using this feature, you can add specific attributes to a table or partition to perform the operations corresponding to the attributes. For example, you can use table attributes to control the Region merge behavior.

Note:

- Currently, TiDB only supports adding the `merge_option` attribute to a table or partition to control the Region merge behavior. The `merge_option` attribute is only part of how to deal with hotspots. For more information, refer to [Troubleshoot Hotspot Issues](#).
- When you use TiDB Binlog or TiCDC to perform replication or use BR to perform incremental backup, the replication or backup operations skip the DDL statement that sets table attributes. To use table attributes in the downstream or in the backup cluster, you need to manually execute the DDL statement in the downstream or in the backup cluster.

11.5.9.1 Usage

The table attribute is in the form of `key=value`. Multiple attributes are separated by commas. In the following examples, `t` is the name of the table to be modified, `p` is the name of the partition to be modified. Items in `[]` are optional.

- Set attributes for a table or partition:

```
ALTER TABLE t [PARTITION p] ATTRIBUTES [=] 'key=value[, key1=value1...]
→ ';
```

- Reset attributes for a table or partition:

```
ALTER TABLE t [PARTITION p] ATTRIBUTES [=] DEFAULT;
```

- See the attributes of all tables and partitions:

```
SELECT * FROM information_schema.attributes;
```

- See the attribute configured to a table or partition:

```
SELECT * FROM information_schema.attributes WHERE id='schema/t[/p]';
```

- See all tables and partitions that have a specific attribute:

```
SELECT * FROM information_schema.attributes WHERE attributes LIKE '%
→ key%';
```

11.5.9.2 Attribute override rules

The attribute configured to a table takes effect on all partitions of the table. However, there is one exception: If the table and partition are configured with the same attribute but different attribute values, the partition attribute overrides the table attribute. For example, suppose that the table `t` is configured with the `key=value` attribute, and the partition `p` is configured with `key=value1`.

```
ALTER TABLE t ATTRIBUTES [=] 'key=value';
ALTER TABLE t PARTITION p ATTRIBUTES [=] 'key=value1';
```

In this case, `key=value1` is the attribute that actually takes effect on the `p1` partition.

11.5.9.3 Control the Region merge behavior using table attributes

11.5.9.3.1 User scenarios

If there is a write hotspot or read hotspot, you can use table attributes to control the Region merge behavior. You can first add the `merge_option` attribute to a table or partition and then set its value to `deny`. The two scenarios are as follows.

Write hotspot on a newly created table or partition

If a hotspot issue occurs when data is written to a newly created table or partition, you usually need to split and scatter Regions. However, if there is a certain time interval between the split/scatter operation and writes, these operations do not truly avoid the write hotspot. This is because the split operation performed when the table or partition is created produces empty Regions, so if the time interval exists, the split Regions might be merged. To handle this case, you can add the `merge_option` attribute to the table or partition and set the attribute value to `deny`.

Periodic read hotspot in read-only scenarios

Suppose that in a read-only scenario, you try to reduce the periodic read hotspot that occurs on a table or partition by manually splitting Regions, and you do not want the manually split Regions to be merged after the hotspot issue is resolved. In this case, you can add the `merge_option` attribute to the table or partition and set its value to `deny`.

11.5.9.3.2 Usage

- Prevent the Regions of a table from merging:

```
ALTER TABLE t ATTRIBUTES 'merge_option=deny';
```

- Allow merging Regions belonging to a table:

```
ALTER TABLE t ATTRIBUTES 'merge_option=allow';
```

- Reset attributes of a table:

```
ALTER TABLE t ATTRIBUTES DEFAULT;
```

- Prevent the Regions of a partition from merging:

```
ALTER TABLE t PARTITION p ATTRIBUTES 'merge_option=deny';
```

- Allow merging Regions belonging to a partition:

```
ALTER TABLE t PARTITION p ATTRIBUTES 'merge_option=allow';
```

- See all tables or partitions configured the `merge_option` attribution:

```
SELECT * FROM information_schema.attributes WHERE attributes LIKE '%
→ merge_option%';
```

11.5.9.3.3 Attribute override rules

```
ALTER TABLE t ATTRIBUTES 'merge_option=deny';
ALTER TABLE t PARTITION p ATTRIBUTES 'merge_option=allow';
```

When the above two attributes are configured at the same time, the Regions belonging to the partition p can actually be merged. When the attribute of the partition is reset, the partition p inherits the attribute from the table t, and the Regions cannot be merged.

Note:

- When there is now only an attribute of a partition, even if the `merge_option=allow` attribute is configured, the partition is still split into multiple Regions by default according to the actual number of partitions. To merge all Regions, you need to [reset the attribute of the table](#).
- When using the `merge_option` attribute, you need to pay attention to the PD configuration parameter `split-merge-interval`. Suppose that the `merge_option` attribute is not configured. In this case, if Regions meet conditions, Regions can be merged after the interval specified by `split-merge-interval`. If the `merge_option` attribute is configured, PD decides whether to merge Regions after the specified interval according to the `merge_option` configuration.

11.5.10 Transactions

11.5.10.1 Transactions

TiDB supports distributed transactions using either [pessimistic](#) or [optimistic](#) transaction mode. Starting from TiDB 3.0.8, TiDB uses the pessimistic transaction mode by default.

This document introduces commonly used transaction-related statements, explicit and implicit transactions, isolation levels, lazy check for constraints, and transaction sizes.

The common variables include `autocommit`, `tidb_disable_txn_auto_retry`, `tidb_retry_limit`, and `tidb_txn_mode`.

Note:

The `tidb_disable_txn_auto_retry` and `tidb_retry_limit` variables only apply to optimistic transactions, not to pessimistic transactions.

11.5.10.1.1 Common statements

Starting a transaction

The statements `BEGIN` and `START TRANSACTION` can be used interchangeably to explicitly start a new transaction.

Syntax:

```
BEGIN;
```

```
START TRANSACTION;
```

```
START TRANSACTION WITH CONSISTENT SNAPSHOT;
```

```
START TRANSACTION WITH CAUSAL CONSISTENCY ONLY;
```

If the current session is in the process of a transaction when one of these statements is executed, TiDB automatically commits the current transaction before starting a new transaction.

Note:

Unlike MySQL, TiDB takes a snapshot of the current database after executing the statements above. MySQL's `BEGIN` and `START TRANSACTION` take a snapshot after executing the first `SELECT` statement (not `SELECT FOR UPDATE` →) that reads data from InnoDB after a transaction is started. `START TRANSACTION WITH CONSISTENT SNAPSHOT` takes a snapshot during the execution of the statement. As a result, `BEGIN`, `START TRANSACTION`, and `START TRANSACTION WITH CONSISTENT SNAPSHOT` are equivalent to `START TRANSACTION WITH CONSISTENT SNAPSHOT` in MySQL.

Committing a transaction

The statement `COMMIT` instructs TiDB to apply all changes made in the current transaction.

Syntax:

```
COMMIT;
```

Tip:

Make sure that your application correctly handles that a `COMMIT` statement could return an error before enabling [optimistic transactions](#). If you are unsure of how your application handles this, it is recommended to instead use the default of [pessimistic transactions](#).

Rolling back a transaction

The statement `ROLLBACK` rolls back and cancels all changes in the current transaction.

Syntax:

```
ROLLBACK;
```

Transactions are also automatically rolled back if the client connection is aborted or closed.

11.5.10.1.2 Autocommit

As required for MySQL compatibility, TiDB will by default *autocommit* statements immediately following their execution.

For example:

```
mysql> CREATE TABLE t1 (
    -> id INT NOT NULL PRIMARY KEY auto_increment,
    -> pad1 VARCHAR(100)
    -> );
Query OK, 0 rows affected (0.09 sec)

mysql> SELECT @@autocommit;
+-----+
| @@autocommit |
+-----+
| 1           |
+-----+
1 row in set (0.00 sec)

mysql> INSERT INTO t1 VALUES (1, 'test');
Query OK, 1 row affected (0.02 sec)

mysql> ROLLBACK;
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT * FROM t1;
```

```
+----+-----+
| id | pad1 |
+----+-----+
| 1  | test |
+----+-----+
1 row in set (0.00 sec)
```

In the above example, the `ROLLBACK` statement has no effect. This is because the `INSERT` statement is executed in autocommit. That is, it was the equivalent of the following single-statement transaction:

```
START TRANSACTION;
INSERT INTO t1 VALUES (1, 'test');
COMMIT;
```

Autocommit will not apply if a transaction has been explicitly started. In the following example, the `ROLLBACK` statement successfully reverts the `INSERT` statement:

```
mysql> CREATE TABLE t2 (
    -> id INT NOT NULL PRIMARY KEY auto_increment,
    -> pad1 VARCHAR(100)
    -> );
Query OK, 0 rows affected (0.10 sec)

mysql> SELECT @@autocommit;
+-----+
| @@autocommit |
+-----+
| 1           |
+-----+
1 row in set (0.00 sec)

mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO t2 VALUES (1, 'test');
Query OK, 1 row affected (0.02 sec)

mysql> ROLLBACK;
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT * FROM t2;
Empty set (0.00 sec)
```

The `autocommit` system variable can be changed on either a global or session basis.

For example:

```
SET autocommit = 0;
```

```
SET GLOBAL autocommit = 0;
```

11.5.10.1.3 Explicit and implicit transaction

Note:

Some statements are committed implicitly. For example, executing [BEGIN|
→ START TRANSACTION] implicitly commits the last transaction and starts
a new transaction. This behavior is required for MySQL compatibility. Refer
to [implicit commit](#) for more details.

TiDB supports explicit transactions (use [BEGIN|START TRANSACTION] and COMMIT to define the start and end of the transaction) and implicit transactions (SET autocommit = 1).

If you set the value of autocommit to 1 and start a new transaction through the [BEGIN|
→ START TRANSACTION] statement, the autocommit is disabled before COMMIT or ROLLBACK
which makes the transaction becomes explicit.

For DDL statements, the transaction is committed automatically and does not support rollback. If you run the DDL statement while the current session is in the process of a transaction, the DDL statement is executed after the current transaction is committed.

11.5.10.1.4 Lazy check of constraints

By default, optimistic transactions will not check the [primary key](#) or [unique constraints](#) when a DML statement is executed. These checks are instead performed on transaction COMMIT.

For example:

```
CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY);
INSERT INTO t1 VALUES (1);
BEGIN OPTIMISTIC;
INSERT INTO t1 VALUES (1); -- MySQL returns an error; TiDB returns success
                           .
INSERT INTO t1 VALUES (2);
COMMIT; -- It is successfully committed in MySQL; TiDB returns an error
        and the transaction rolls back.
SELECT * FROM t1; -- MySQL returns 1 2; TiDB returns 1.
```

```

mysql> CREATE TABLE t1 (id INT NOT NULL PRIMARY KEY);
Query OK, 0 rows affected (0.10 sec)

mysql> INSERT INTO t1 VALUES (1);
Query OK, 1 row affected (0.02 sec)

mysql> BEGIN OPTIMISTIC;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO t1 VALUES (1); -- MySQL returns an error; TiDB returns
     → success.
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO t1 VALUES (2);
Query OK, 1 row affected (0.00 sec)

mysql> COMMIT; -- It is successfully committed in MySQL; TiDB returns an
     → error and the transaction rolls back.
ERROR 1062 (23000): Duplicate entry '1' for key 'PRIMARY'
mysql> SELECT * FROM t1; -- MySQL returns 1 2; TiDB returns 1.
+---+
| id |
+---+
| 1 |
+---+
1 row in set (0.01 sec)

```

The lazy check optimization improves performance by batching constraint checks and reducing network communication. The behavior can be disabled by setting `tidb_constraint_check_in_place=TRUE`.

Note:

- This optimization only applies to optimistic transactions.
- This optimization does not take effect for `INSERT IGNORE` and `INSERT → ON DUPLICATE KEY UPDATE`, but only for normal `INSERT` statements.

11.5.10.1.5 Statement rollback

TiDB supports atomic rollback after statement execution failure. If a statement results in an error, the changes it made will not take effect. The transaction will remain open, and

additional changes can be made before issuing a COMMIT or ROLLBACK statement.

```
CREATE TABLE test (id INT NOT NULL PRIMARY KEY);
BEGIN;
INSERT INTO test VALUES (1);
INSERT INTO tset VALUES (2); -- Statement does not take effect because "
    ↪ "test" is misspelled as "tset".
INSERT INTO test VALUES (1),(2); -- Entire statement does not take effect
    ↪ because it violates a PRIMARY KEY constraint
INSERT INTO test VALUES (3);
COMMIT;
SELECT * FROM test;
```

```
mysql> CREATE TABLE test (id INT NOT NULL PRIMARY KEY);
Query OK, 0 rows affected (0.09 sec)

mysql> BEGIN;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO test VALUES (1);
Query OK, 1 row affected (0.02 sec)

mysql> INSERT INTO tset VALUES (2); -- Statement does not take effect
    ↪ because "test" is misspelled as "tset".
ERROR 1146 (42S02): Table 'test.tset' doesn't exist
mysql> INSERT INTO test VALUES (1),(2); -- Entire statement does not take
    ↪ effect because it violates a PRIMARY KEY constraint
ERROR 1062 (23000): Duplicate entry '1' for key 'PRIMARY'
mysql> INSERT INTO test VALUES (3);
Query OK, 1 row affected (0.00 sec)

mysql> COMMIT;
Query OK, 0 rows affected (0.01 sec)

mysql> SELECT * FROM test;
+---+
| id |
+---+
| 1 |
| 3 |
+---+
2 rows in set (0.00 sec)
```

In the above example, the transaction remains open after the failed INSERT statements. The final insert statement is then successful and changes are committed.

11.5.10.1.6 Transaction size limit

Due to the limitations of the underlying storage engine, TiDB requires a single row to be no more than 6 MB. All columns of a row are converted to bytes according to their data types and summed up to estimate the size of a single row.

TiDB supports both optimistic and pessimistic transactions, and optimistic transactions are the basis for pessimistic transactions. Because optimistic transactions first cache the changes in private memory, TiDB limits the size of a single transaction.

By default, TiDB sets the total size of a single transaction to no more than 100 MB. You can modify this default value via `txn-total-size-limit` in the configuration file. The maximum value of `txn-total-size-limit` is 10 GB.

The actual individual transaction size limit also depends on the amount of remaining memory available to the server, because when a transaction is executed, the memory usage of the TiDB process is approximately six times the size of the transaction.

TiDB previously limited the total number of key-value pairs for a single transaction to 300,000. This restriction was removed in TiDB v4.0.

Note:

Usually, TiDB Binlog is enabled to replicate data to the downstream. In some scenarios, message middleware such as Kafka is used to consume binlogs that are replicated to the downstream.

Taking Kafka as an example, the upper limit of Kafka's single message processing capability is 1 GB. Therefore, when `txn-total-size-limit` is set to more than 1 GB, it might happen that the transaction is successfully executed in TiDB, but the downstream Kafka reports an error. To avoid this situation, you need to decide the actual value of `txn-total-size-limit` according to the limit of the end consumer. For example, if Kafka is used downstream, `txn-total-size-limit` must not exceed 1 GB.

11.5.10.1.7 Causal consistency

Note:

Transactions with causal consistency take effect only when the async commit and one-phase commit features are enabled. For details of the two features, see `tidb_enable_async_commit` and `tidb_enable_1pc`.

TiDB supports enabling causal consistency for transactions. Transactions with causal consistency, when committed, do not need to get timestamp from PD and have lower commit latency. The syntax to enable causal consistency is as follows:

```
START TRANSACTION WITH CAUSAL CONSISTENCY ONLY;
```

By default, TiDB guarantees linear consistency. In the case of linear consistency, if transaction 2 is committed after transaction 1 is committed, logically, transaction 2 should occur after transaction 1. Causal consistency is weaker than linear consistency. In the case of causal consistency, the commit order and occurrence order of two transactions can be guaranteed consistent only when the data locked or written by transaction 1 and transaction 2 have an intersection, which means that the two transactions have a causal relationship known to the database. Currently, TiDB does not support passing in external causal relationship.

Two transactions with causal consistency enabled have the following characteristics:

- Transactions with potential causal relationship have the consistent logical order and physical commit order
- Transactions with no causal relationship do not guarantee consistent logical order and physical commit order
- Reads without lock do not create causal relationship

Transactions with potential causal relationship have the consistent logical order and physical commit order

Assume that both transaction 1 and transaction 2 adopt causal consistency and have the following statements executed:

Transaction 1	Transaction 2
1	2
START	START
TRANS-	TRANS-
AC-	AC-
TION	TION
WITH	WITH
CAUSAL	CAUSAL
CON-	CON-
SIS-	SIS-
TENCY	TENCY
ONLY	ONLY

Transaction	Transaction
1	2
<hr/>	
x = SE-	
LECT v	
FROM	
t	
WHERE	
id = 1	
FOR	
UP-	
DATE	
UPDATE	
t set v	
=	
\$(x + 1)	
WHERE	
id = 2	
COMMIT	
UPDATE	
t SET v	
= 2	
WHERE	
id = 1	
COMMIT	

In the example above, transaction 1 locks the `id = 1` record and transaction 2 modifies the `id = 1` record. Therefore, transaction 1 and transaction 2 have a potential causal relationship. Even with the causal consistency enabled, as long as transaction 2 is committed after transaction 1 is successfully committed, logically, transaction 2 must occur after transaction 1. Therefore, it is impossible that a transaction reads transaction 2's modification on the `id = 1` record without reading transaction 1's modification on the `id = 2` record.

Transactions with no causal relationship do not guarantee consistent logical order and physical commit order

Assume that the initial values of `id = 1` and `id = 2` are both 0. Assume that both transaction 1 and transaction 2 adopt causal consistency and have the following statements executed:

Transaction 1	Transaction 2	Transaction 3
1	2	3
START	START	
TRANS-	TRANS-	
AC-	AC-	
TION	TION	
WITH	WITH	
CAUSAL	CAUSAL	
CON-	CON-	
SIS-	SIS-	
TENCY	TENCY	
ONLY	ONLY	
UPDATE		
t set v		
= 3		
WHERE		
id = 2		
	UPDATE	
	t SET v	
	= 2	
	WHERE	
	id = 1	
	BEGIN	
COMMIT		
	COMMIT	
		SELECT
		v
		FROM
		t
		WHERE
		id IN
		(1, 2)

In the example above, transaction 1 does not read the `id = 1` record, so transaction 1 and transaction 2 have no causal relationship known to the database. With causal consistency enabled for the transactions, even if transaction 2 is committed after transaction 1 is committed in terms of physical time order, TiDB does not guarantee that transaction 2 logically occurs after transaction 1.

If transaction 3 begins before transaction 1 is committed, and if transaction 3 reads the `id = 1` and `id = 2` records after transaction 2 is committed, transaction 3 might read the value of `id = 1` to be 2 but the value of `id = 2` to be 0.

Reads without lock do not create causal relationship

Assume that both transaction 1 and transaction 2 adopt causal consistency and have

the following statements executed:

Transaction 1	Transaction 2
START	START
TRANS-	TRANS-
AC-	AC-
TION	TION
WITH	WITH
CAUSAL	CAUSAL
CON-	CON-
SIS-	SIS-
TENCY	TENCY
ONLY	ONLY
	UPDATE
	t SET v
	= 2
	WHERE
	id = 1
SELECT	
v	
FROM	
t	
WHERE	
id = 1	
UPDATE	
t set v	
= 3	
WHERE	
id = 2	
	COMMIT
	COMMIT

In the example above, reads without lock do not create causal relationship. Transaction 1 and transaction 2 have created write skew. In this case, it would have been unreasonable if the two transactions still had causal relationship. Therefore, the two transactions with causal consistency enabled have no definite logical order.

11.5.10.2 TiDB Transaction Isolation Levels

Transaction isolation is one of the foundations of database transaction processing. Isolation is one of the four key properties of a transaction (commonly referred as **ACID**).

The SQL-92 standard defines four levels of transaction isolation: Read Uncommitted, Read Committed, Repeatable Read, and Serializable. See the following table for details:

Isolation Level	Dirty Write	Dirty Read	Fuzzy Read	Phantom
READ UNCOMMITTED	Not Possible	Possible	Possible	Possible
READ COMMITTED	Not Possible	Not possible	Possible	Possible
REPEATABLE READ	Not Possible	Not possible	Not possible	Possible
SERIALIZABLE	Not Possible	Not possible	Not possible	Not possible

TiDB implements Snapshot Isolation (SI) consistency, which it advertises as REPEATABLE → -READ for compatibility with MySQL. This differs from the [ANSI Repeatable Read isolation level](#) and the [MySQL Repeatable Read level](#).

Note:

In TiDB v3.0, the automatic retry of transactions is disabled by default. It is not recommended to enable the automatic retry because it might **break the transaction isolation level**. Refer to [Transaction Retry](#) for details.

Starting from TiDB [v3.0.8](#), newly created TiDB clusters use the **pessimistic transaction mode** by default. The current read (for update read) is **non-repeatable read**. Refer to [pessimistic transaction mode](#) for details.

11.5.10.2.1 Repeatable Read isolation level

The Repeatable Read isolation level only sees data committed before the transaction begins, and it never sees either uncommitted data or changes committed during transaction execution by concurrent transactions. However, the transaction statement does see the effects of previous updates executed within its own transaction, even though they are not yet committed.

For transactions running on different nodes, the start and commit order depends on the order that the timestamp is obtained from PD.

Transactions of the Repeatable Read isolation level cannot concurrently update a same row. When committing, if the transaction finds that the row has been updated by another transaction after it starts, then the transaction rolls back. For example:

```
create table t1(id int);
insert into t1 values(0);

start transaction;           |
select * from t1;          |
update t1 set id=id+1;     |
→ pessimistic transactions, the `update` statement executed later
→ waits for the lock until the transaction holding the lock commits or
→ rolls back and releases the row lock.

start transaction;
select * from t1;
update t1 set id=id+1; -- In
```

<pre><code>commit;</code></pre>	<pre><code>commit; -- The transaction commit ↪ fails and rolls back. Pessimistic ↪ transactions can commit successfully.</code></pre>
---------------------------------	---

Difference between TiDB and ANSI Repeatable Read

The Repeatable Read isolation level in TiDB differs from ANSI Repeatable Read isolation level, though they share the same name. According to the standard described in the [A Critique of ANSI SQL Isolation Levels](#) paper, TiDB implements the Snapshot Isolation level. This isolation level does not allow strict phantoms (A3) but allows broad phantoms (P3) and write skews. In contrast, the ANSI Repeatable Read isolation level allows phantom reads but does not allow write skews.

Difference between TiDB and MySQL Repeatable Read

The Repeatable Read isolation level in TiDB differs from that in MySQL. The MySQL Repeatable Read isolation level does not check whether the current version is visible when updating, which means it can continue to update even if the row has been updated after the transaction starts. In contrast, if the row has been updated after the transaction starts, the TiDB optimistic transaction is rolled back and retried. Transaction retries in TiDB's optimistic concurrency control might fail, leading to a final failure of the transaction, while in TiDB's pessimistic concurrency control and MySQL, the updating transaction can be successful.

11.5.10.2.2 Read Committed isolation level

Starting from TiDB v4.0.0-beta, TiDB supports the Read Committed isolation level.

For historical reasons, the Read Committed isolation level of current mainstream databases is essentially the [Consistent Read isolation level defined by Oracle](#). In order to adapt to this situation, the Read Committed isolation level in TiDB pessimistic transactions is also a consistent read behavior in essence.

Note:

The Read Committed isolation level only takes effect in the [pessimistic transaction mode](#). In the [optimistic transaction mode](#), setting the transaction isolation level to `Read Committed` does not take effect and transactions still use the Repeatable Read isolation level.

11.5.10.2.3 Difference between TiDB and MySQL Read Committed

The MySQL Read Committed isolation level is in line with the Consistent Read features in most cases. There are also exceptions, such as [semi-consistent read](#). This special behavior is not supported in TiDB.

11.5.10.3 TiDB Optimistic Transaction Model

With optimistic transactions, conflicting changes are detected as part of a transaction commit. This helps improve the performance when concurrent transactions are infrequently modifying the same rows, because the process of acquiring row locks can be skipped. In the case that concurrent transactions frequently modify the same rows (a conflict), optimistic transactions may perform worse than [Pessimistic Transactions](#).

Before enabling optimistic transactions, make sure that your application correctly handles that a `COMMIT` statement could return errors. If you are unsure of how your application handles this, it is recommended to instead use Pessimistic Transactions.

Note:

Starting from v3.0.8, TiDB uses the [pessimistic transaction mode](#) by default. However, this does not affect your existing cluster if you upgrade it from v3.0.7 or earlier to v3.0.8 or later. In other words, **only newly created clusters default to using the pessimistic transaction mode**.

11.5.10.3.1 Principles of optimistic transactions

To support distributed transactions, TiDB adopts two-phase commit (2PC) in optimistic transactions. The procedure is as follows:

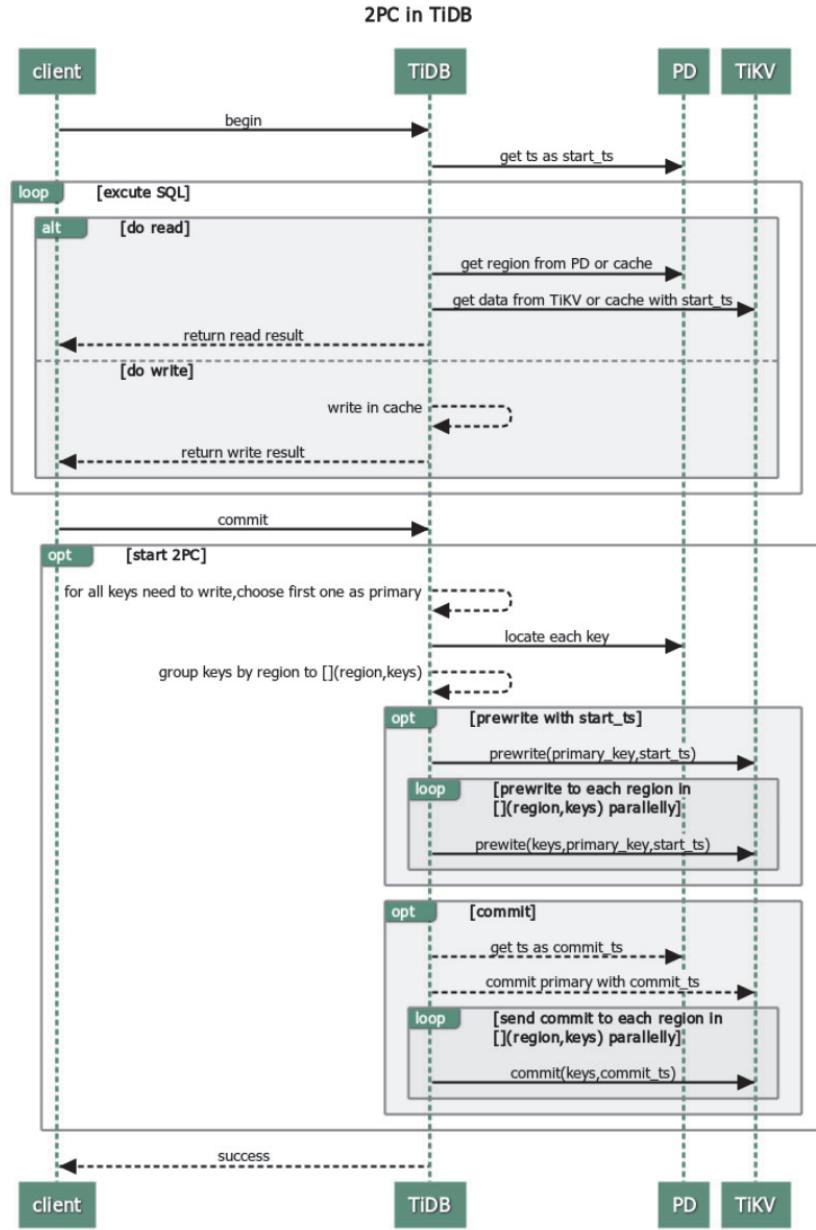


Figure 334: 2PC in TiDB

1. The client begins a transaction.

TiDB gets a timestamp (monotonically increasing in time and globally unique) from PD as the unique transaction ID of the current transaction, which is called `start_ts`. TiDB implements multi-version concurrency control, so `start_ts` also serves as the version of the database snapshot obtained by this transaction. This means that the transaction can only read the data from the database at `start_ts`.

2. The client issues a read request.

1. TiDB receives routing information (how data is distributed among TiKV nodes) from PD.
2. TiDB receives the data of the `start_ts` version from TiKV.
3. The client issues a write request.
TiDB checks whether the written data satisfies constraints (to ensure the data types are correct, the NOT NULL constraint is met, etc.). **Valid data is stored in the private memory of this transaction in TiDB.**
4. The client issues a commit request.
5. TiDB begins 2PC, and persists data in store while guaranteeing the atomicity of transactions.
 1. TiDB selects a Primary Key from the data to be written.
 2. TiDB receives the information of Region distribution from PD, and groups all keys by Region accordingly.
 3. TiDB sends prewrite requests to all TiKV nodes involved. Then, TiKV checks whether there are conflict or expired versions. Valid data is locked.
 4. TiDB receives all responses in the prewrite phase and the prewrite is successful.
 5. TiDB receives a commit version number from PD and marks it as `commit_ts`.
 6. TiDB initiates the second commit to the TiKV node where Primary Key is located. TiKV checks the data, and cleans the locks left in the prewrite phase.
 7. TiDB receives the message that reports the second phase is successfully finished.
6. TiDB returns a message to inform the client that the transaction is successfully committed.
7. TiDB asynchronously cleans the locks left in this transaction.

11.5.10.3.2 Advantages and disadvantages

From the process of transactions in TiDB above, it is clear that TiDB transactions have the following advantages:

- Simple to understand
- Implement cross-node transaction based on single-row transaction
- Decentralized lock management

However, TiDB transactions also have the following disadvantages:

- Transaction latency due to 2PC
- In need of a centralized timestamp allocation service
- OOM (out of memory) when extensive data is written in the memory

11.5.10.3.3 Transaction retries

In the optimistic transaction model, transactions might fail to be committed because of write-write conflict in heavy contention scenarios. TiDB uses optimistic concurrency control by default, whereas MySQL applies pessimistic concurrency control. This means that MySQL adds locks during the execution of write-type SQL statements, and its Repeatable Read isolation level allows for current reads, so commits generally do not encounter exceptions. To lower the difficulty of adapting applications, TiDB provides an internal retry mechanism.

Automatic retry

If a write-write conflict occurs during the transaction commit, TiDB automatically retries the SQL statement that includes write operations. You can enable the automatic retry by setting `tidb_disable_txn_auto_retry` to OFF and set the retry limit by configuring `tidb_retry_limit`:

```
#### Whether to disable automatic retry. ("on" by default)
tidb_disable_txn_auto_retry = OFF
#### Set the maximum number of the retries. ("10" by default)
#### When "tidb_retry_limit = 0" , automatic retry is completely disabled.
tidb_retry_limit = 10
```

You can enable the automatic retry in either session level or global level:

1. Session level:

```
SET tidb_disable_txn_auto_retry = OFF;
```

```
SET tidb_retry_limit = 10;
```

2. Global level:

```
SET GLOBAL tidb_disable_txn_auto_retry = OFF;
```

```
SET GLOBAL tidb_retry_limit = 10;
```

Note:

The `tidb_retry_limit` variable decides the maximum number of retries. When this variable is set to 0, none of the transactions automatically retries, including the implicit single statement transactions that are automatically committed. This is the way to completely disable the automatic retry mechanism in TiDB. After the automatic retry is disabled, all conflicting transactions report failures (including the `try again later` message) to the application layer in the fastest way.

Limits of retry

By default, TiDB will not retry transactions because this might lead to lost updates and damaged `REPEATABLE READ` isolation.

The reason can be observed from the procedures of retry:

1. Allocate a new timestamp and mark it as `start_ts`.
2. Retry the SQL statements that contain write operations.
3. Implement the two-phase commit.

In Step 2, TiDB only retries SQL statements that contain write operations. However, during retrying, TiDB receives a new version number to mark the beginning of the transaction. This means that TiDB retries SQL statements with the data in the new `start_ts` version. In this case, if the transaction updates data using other query results, the results might be inconsistent because the `REPEATABLE READ` isolation is violated.

If your application can tolerate lost updates, and does not require `REPEATABLE READ` isolation consistency, you can enable this feature by setting `tidb_disable_txn_auto_retry ↴ = OFF`.

11.5.10.3.4 Conflict detection

As a distributed database, TiDB performs in-memory conflict detection in the TiKV layer, mainly in the prewrite phase. TiDB instances are stateless and unaware of each other, which means they cannot know whether their writes result in conflicts across the cluster. Therefore, conflict detection is performed in the TiKV layer.

The configuration is as follows:

```
#### Controls the number of slots. ("2048000" by default)
scheduler-concurrency = 2048000
```

In addition, TiKV supports monitoring the time spent on waiting latches in the scheduler.

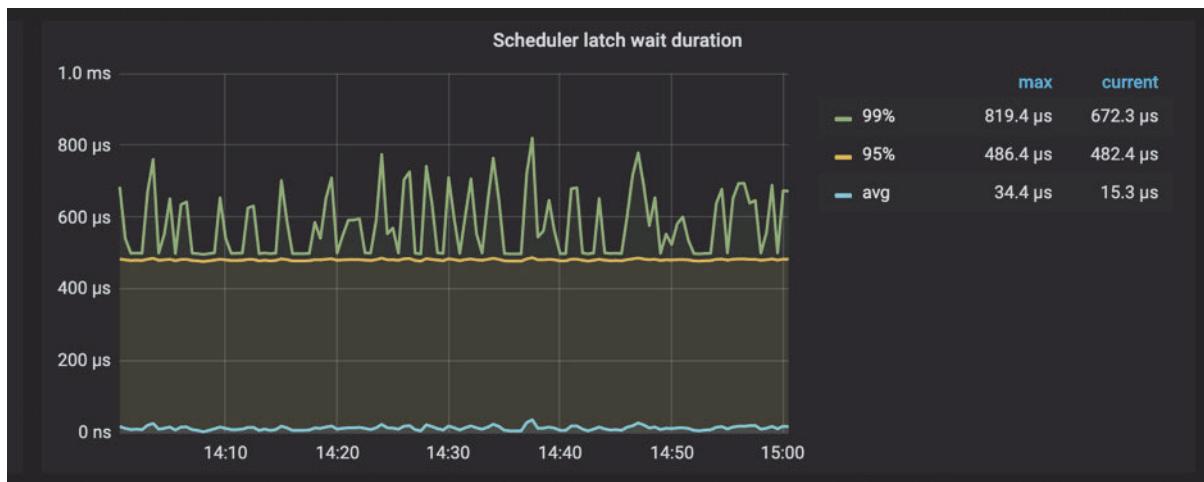


Figure 335: Scheduler latch wait duration

When Scheduler latch wait duration is high and there are no slow writes, it can be safely concluded that there are many write conflicts at this time.

11.5.10.4 TiDB Pessimistic Transaction Mode

To make the usage of TiDB closer to traditional databases and reduce the cost of migration, starting from v3.0, TiDB supports the pessimistic transaction mode on top of the optimistic transaction model. This document describes the features of the TiDB pessimistic transaction mode.

Note:

Starting from v3.0.8, newly created TiDB clusters use the pessimistic transaction mode by default. However, this does not affect your existing cluster if you upgrade it from v3.0.7 or earlier to v3.0.8 or later. In other words, **only newly created clusters default to using the pessimistic transaction mode.**

11.5.10.4.1 Switch transaction mode

You can set the transaction mode by configuring the `tidb_txn_mode` system variable. The following command sets all explicit transactions (that is, non-autocommit transactions) executed by newly created sessions in the cluster to the pessimistic transaction mode:

```
SET GLOBAL tidb_txn_mode = 'pessimistic';
```

You can also explicitly enable the pessimistic transaction mode by executing the following SQL statements:

```
BEGIN PESSIMISTIC;
```

```
BEGIN /*T! PESSIMISTIC */;
```

The `BEGIN PESSIMISTIC;` and `BEGIN OPTIMISTIC;` statements take precedence over the `tidb_txn_mode` system variable. Transactions started with these two statements ignore the system variable and support using both the pessimistic and optimistic transaction modes.

11.5.10.4.2 Behaviors

Pessimistic transactions in TiDB behave similarly to those in MySQL. See the minor differences in [Difference with MySQL InnoDB](#).

- When you execute `UPDATE`, `DELETE` or `INSERT` statements, the **latest** committed data is read, data is modified, and a pessimistic lock is applied on the modified rows.

- For `SELECT FOR UPDATE` statements, a pessimistic lock is applied on the latest version of the committed data, instead of on the modified rows.
- Locks will be released when the transaction is committed or rolled back. Other transactions attempting to modify the data are blocked and have to wait for the lock to be released. Transactions attempting to *read* the data are not blocked, because TiDB uses multi-version concurrency control (MVCC).
- If several transactions are trying to acquire each other's respective locks, a deadlock will occur. This is automatically detected, and one of the transactions will randomly be terminated with a MySQL-compatible error code 1213 returned.
- Transactions will wait up to `innodb_lock_wait_timeout` seconds (default: 50) to acquire new locks. When this timeout is reached, a MySQL-compatible error code 1205 is returned. If multiple transactions are waiting for the same lock, the order of priority is approximately based on the `start ts` of the transaction.
- TiDB supports both the optimistic transaction mode and pessimistic transaction mode in the same cluster. You can specify either mode for transaction execution.
- TiDB supports the `FOR UPDATE NOWAIT` syntax and does not block and wait for locks to be released. Instead, a MySQL-compatible error code 3572 is returned.
- If the `Point Get` and `Batch Point Get` operators do not read data, they still lock the given primary key or unique key, which blocks other transactions from locking or writing data to the same primary key or unique key.
- TiDB supports the `FOR UPDATE OF TABLES` syntax. For a statement that joins multiple tables, TiDB only applies pessimistic locks on the rows that are associated with the tables in `OF TABLES`.

11.5.10.4.3 Difference with MySQL InnoDB

1. When TiDB executes DML or `SELECT FOR UPDATE` statements that use range in the `WHERE` clause, concurrent DML statements within the range are not blocked.

For example:

```
CREATE TABLE t1 (
    id INT NOT NULL PRIMARY KEY,
    pad1 VARCHAR(100)
);
INSERT INTO t1 (id) VALUES (1),(5),(10);
```

```
BEGIN /*T! PESSIMISTIC */;
SELECT * FROM t1 WHERE id BETWEEN 1 AND 10 FOR UPDATE;
```

```
BEGIN /*T! PESSIMISTIC */;
INSERT INTO t1 (id) VALUES (6); -- blocks only in MySQL
UPDATE t1 SET pad1='new value' WHERE id = 5; -- blocks waiting in both
    ↪ MySQL and TiDB
```

This behavior is because TiDB does not currently support *gap locking*.

2. TiDB does not support `SELECT LOCK IN SHARE MODE`.

When `SELECT LOCK IN SHARE MODE` is executed, it has the same effect as that without the lock, so the read or write operation of other transactions is not blocked.

3. DDL may result in failure of the pessimistic transaction commit.

When DDL is executed in MySQL, it might be blocked by the transaction that is being executed. However, in this scenario, the DDL operation is not blocked in TiDB, which leads to failure of the pessimistic transaction commit: `ERROR 1105 (HY000) ↪ : Information schema is changed. [try again later]`. TiDB executes the `TRUNCATE TABLE` statement during the transaction execution, which might result in the `table doesn't exist` error.

4. After executing `START TRANSACTION WITH CONSISTENT SNAPSHOT`, MySQL can still read the tables that are created later in other transactions, while TiDB cannot.
5. The autocommit transactions prefer the optimistic locking.

When using the pessimistic model, the autocommit transactions first try to commit the statement using the optimistic model that has less overhead. If a write conflict occurs, the pessimistic model is used for transaction retry. Therefore, if `tidb_retry_limit` is set to 0, the autocommit transaction still reports the `Write Conflict` error when a write conflict occurs.

The autocommit `SELECT FOR UPDATE` statement does not wait for lock.

6. The data read by `EMBEDDED SELECT` in the statement is not locked.
7. Open transactions in TiDB do not block garbage collection (GC). By default, this limits the maximum execution time of pessimistic transactions to 1 hour. You can modify this limit by editing `max-txn-ttl` under `[performance]` in the TiDB configuration file.

11.5.10.4.4 Isolation level

TiDB supports the following two isolation levels in the pessimistic transaction mode:

- **Repeatable Read** by default, which is the same as MySQL.

Note:

In this isolation level, DML operations are performed based on the latest committed data. The behavior is the same as MySQL, but differs from the optimistic transaction mode in TiDB. See [Difference between TiDB and MySQL Repeatable Read](#).

- **Read Committed.** You can set this isolation level using the `SET TRANSACTION` statement.

11.5.10.4.5 Pipelined locking process

Adding a pessimistic lock requires writing data into TiKV. The response of successfully adding a lock can only be returned to TiDB after commit and apply through Raft. Therefore, compared with optimistic transactions, the pessimistic transaction mode inevitably has higher latency.

To reduce the overhead of locking, TiKV implements the pipelined locking process: when the data meets the requirements for locking, TiKV immediately notifies TiDB to execute subsequent requests and writes into the pessimistic lock asynchronously. This process reduces most latency and significantly improves the performance of pessimistic transactions. However, when network partition occurs in TiKV or a TiKV node is down, the asynchronous write into the pessimistic lock might fail and affect the following aspects:

- Other transactions that modify the same data cannot be blocked. If the application logic relies on locking or lock waiting mechanisms, the correctness of the application logic is affected.
- There is a low probability that the transaction commit fails, but it does not affect the correctness of the transactions.

If the application logic relies on the locking or lock waiting mechanisms, or if you want to guarantee as much as possible the success rate of transaction commits even in the case of TiKV cluster anomalies, you should disable the pipelined locking feature.

pipelined pessimistic lock

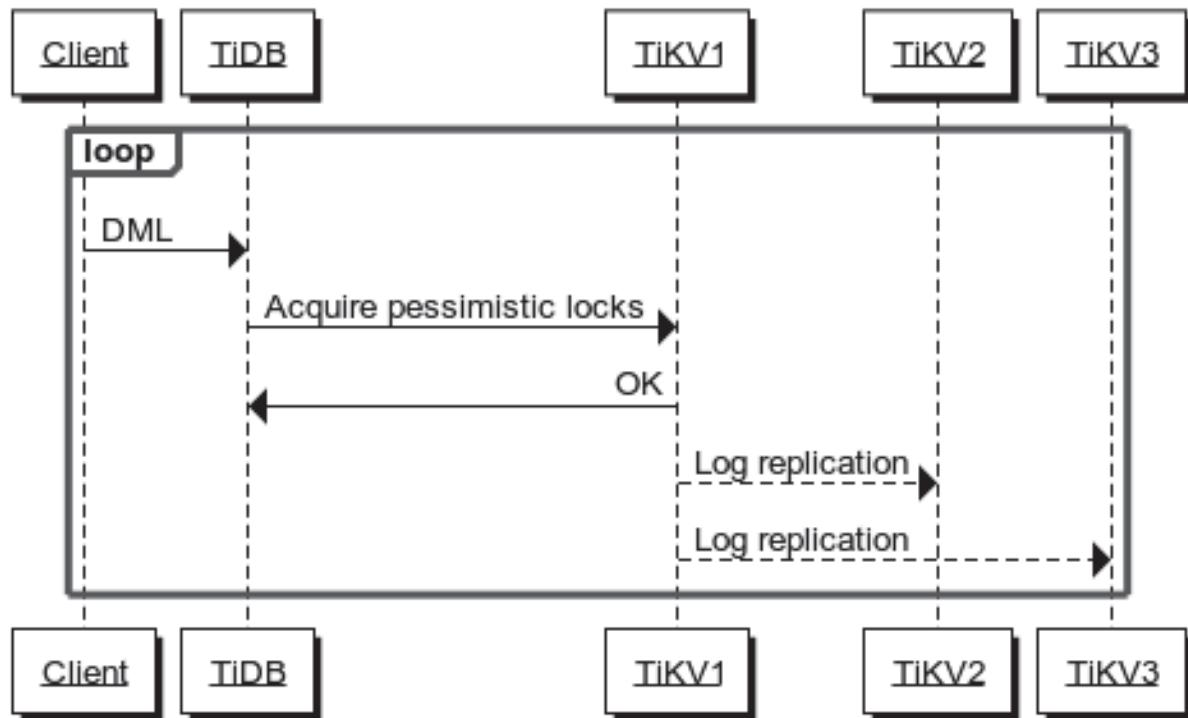


Figure 336: Pipelined pessimistic lock

This feature is enabled by default. To disable it, modify the TiKV configuration:

```
[pessimistic-txn]
pipelined = false
```

If the TiKV cluster is v4.0.9 or later, you can also dynamically disable this feature by modifying TiKV configuration online:

```
set config tikv pessimistic-txn.pipelined='false';
```

11.5.11 Garbage Collection (GC)

11.5.11.1 GC Overview

TiDB uses MVCC to control transaction concurrency. When you update the data, the original data is not deleted immediately but is kept together with the new data, with a timestamp to distinguish the version. The goal of Garbage Collection (GC) is to clear the obsolete data.

11.5.11.1.1 GC process

Each TiDB cluster contains a TiDB instance that is selected as the GC leader, which controls the GC process.

GC runs periodically on TiDB. For each GC, TiDB firstly calculates a timestamp called “safe point”. Then, TiDB clears the obsolete data under the premise that all the snapshots after the safe point retain the integrity of the data. Specifically, there are three steps involved in each GC process:

1. Resolve Locks. During this step, TiDB scans locks before the safe point on all Regions and clears these locks.
2. Delete Ranges. During this step, the obsolete data of the entire range generated from the `DROP TABLE/DROP INDEX` operation is quickly cleared.
3. Do GC. During this step, each TiKV node scans data on it and deletes unneeded old versions of each key.

In the default configuration, GC is triggered every 10 minutes. Each GC retains data of the recent 10 minutes, which means that the GC life time is 10 minutes by default (safe point = the current time - GC life time). If one round of GC has been running for too long, before this round of GC is completed, the next round of GC will not start even if it is time to trigger the next GC. In addition, for long-duration transactions to run properly after exceeding the GC life time, the safe point does not exceed the start time (`start_ts`) of the ongoing transactions.

11.5.11.1.2 Implementation details

Resolve Locks

The TiDB transaction model is implemented based on [Google's Percolator](#). It's mainly a two-phase commit protocol with some practical optimizations. When the first phase is finished, all the related keys are locked. Among these locks, one is the primary lock and the others are secondary locks which contain a pointer to the primary lock; in the second phase, the key with the primary lock gets a write record and its lock is removed. The write record indicates the write or delete operation in the history or the transactional rollback record of this key. The type of write record that replaces the primary lock indicates whether the corresponding transaction is committed successfully. Then all the secondary locks are replaced successively. If, for some reason such as failure, these secondary locks are retained and not replaced, you can still find the primary key based on the information in the secondary locks and determines whether the entire transaction is committed based on whether the primary key is committed. However, if the primary key information is cleared by GC and this transaction has uncommitted secondary locks, you will never learn whether these locks can be committed. As a result, data integrity cannot be guaranteed.

The Resolve Locks step clears the locks before the safe point. This means that if the primary key of a lock is committed, this lock needs to be committed; otherwise, it needs

to be rolled back. If the primary key is still locked (not committed or rolled back), this transaction is seen as timing out and rolled back.

The Resolve Locks step is implemented in either of the following two ways, which can be configured using the system variable `tidb_gc_scan_lock_mode`:

Warning:

Currently, PHYSICAL (Green GC) is an experimental feature. It is not recommended that you use it in the production environment.

- **LEGACY** (default): The GC leader sends requests to all Regions to scan obsolete locks, checks the primary key statuses of scanned locks, and sends requests to commit or roll back the corresponding transaction.
- **PHYSICAL**: TiDB bypasses the Raft layer and directly scans data on each TiKV node.

Delete Ranges

A great amount of data with consecutive keys is removed during operations such as `DROP TABLE/INDEX`. Removing each key and performing GC later for them can result in low execution efficiency on storage reclaiming. In such scenarios, TiDB actually does not delete each key. Instead, it only records the range to be removed and the timestamp of the deletion. Then the Delete Ranges step performs a fast physical deletion on the ranges whose timestamp is before the safe point.

Do GC

The Do GC step clears the outdated versions for all keys. To guarantee that all timestamps after the safe point have consistent snapshots, this step deletes the data committed before the safe point, but retains the last write for each key before the safe point as long as it is not a deletion.

In this step, TiDB only needs to send the safe point to PD, and then the whole round of GC is completed. TiKV automatically detects the change of safe point and performs GC for all Region leaders on the current node. At the same time, the GC leader can continue to trigger the next round of GC.

Note:

Starting with TiDB 5.0, the Do GC step will always use the `DISTRIBUTED gc` mode. This replaces the earlier `CENTRAL gc` mode, which was implemented by TiDB servers sending GC requests to each Region.

11.5.11.2 Garbage Collection Configuration

Garbage collection is configured via the following system variables:

- `tidb_gc_enable`
- `tidb_gc_run_interval`
- `tidb_gc_life_time`
- `tidb_gc_concurrency`
- `tidb_gc_scan_lock_mode`

11.5.11.2.1 GC I/O limit

TiKV supports the GC I/O limit. You can configure `gc.max-write-bytes-per-sec` to limit writes of a GC worker per second, and thus to reduce the impact on normal requests.

0 indicates disabling this feature.

You can dynamically modify this configuration using tikv-ctl:

```
tikv-ctl --host=ip:port modify-tikv-config -m server -n gc.
  ↪ max_write_bytes_per_sec -v 10MB
```

11.5.11.2.2 Changes in TiDB 5.0

In previous releases of TiDB, garbage collection was configured via the `mysql.tidb` system table. While changes to this table continue to be supported, it is recommended to use the system variables provided. This helps ensure that any changes to configuration can be validated, and prevent unexpected behavior ([#20655](#)).

The `CENTRAL` garbage collection mode is no longer supported. The `DISTRIBUTED` GC mode (which has been the default since TiDB 3.0) will automatically be used in its place. This mode is more efficient, since TiDB no longer needs to send requests to each TiKV region to trigger garbage collection.

For information on changes in previous releases, refer to earlier versions of this document using the *TIDB version selector* in the left hand menu.

GC in Compaction Filter

Based on the `DISTRIBUTED` GC mode, the mechanism of GC in Compaction Filter uses the compaction process of RocksDB, instead of a separate GC worker thread, to run GC. This new GC mechanism helps to avoid extra disk read caused by GC. Also, after clearing the obsolete data, it avoids a large number of left tombstone marks which degrade the sequential scan performance. The following example shows how to enable the mechanism in the TiKV configuration file:

```
[gc]
enable-compaction-filter = true
```

You can also enable this GC mechanism by modifying the configuration online. See the following example:

```
show config where type = 'tikv' and name like '%enable-compaction-filter%';
```

Type	Instance	Name	Value
tikv	172.16.5.37:20163	gc.enable-compaction-filter	false
tikv	172.16.5.36:20163	gc.enable-compaction-filter	false
tikv	172.16.5.35:20163	gc.enable-compaction-filter	false

```
set config tikv gc.enable-compaction-filter = true;
show config where type = 'tikv' and name like '%enable-compaction-filter%';
```

Type	Instance	Name	Value
tikv	172.16.5.37:20163	gc.enable-compaction-filter	true
tikv	172.16.5.36:20163	gc.enable-compaction-filter	true
tikv	172.16.5.35:20163	gc.enable-compaction-filter	true

11.5.12 Views

TiDB supports views. A view acts as a virtual table, whose schema is defined by the SELECT statement that creates the view. Using views has the following benefits:

- Exposing only safe fields and data to users to ensure security of sensitive fields and data stored in the underlying table.
- Defining complex queries that frequently appear as views to make complex queries easier and more convenient.

11.5.12.1 Query views

Querying a view is similar to querying an ordinary table. However, when TiDB queries a view, it actually queries the SELECT statement associated with the view.

11.5.12.2 Show metadata

To obtain the metadata of views, choose any of the following methods.

11.5.12.2.1 Use the SHOW CREATE TABLE view_name or SHOW CREATE VIEW view_name statement

Usage example:

```
show create view v;
```

This statement shows the CREATE VIEW statement corresponding to this view and the value of the character_set_client and collation_connection system variables when the view was created.

```
+--+
→ -----
→
| View | Create View
→
→ | character_set_client | collation_connection |
+--+
→ -----
→
| v   | CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`127.0.0.1` SQL SECURITY
→ DEFINER VIEW `v` (`a`) AS SELECT `s`.`a` FROM `test`.`t` LEFT JOIN `test`.`s` ON `t`.`a`=`s`.`a` | utf8 | utf8_general_ci |
+--+
→ -----
→
1 row in set (0.00 sec)
```

11.5.12.2.2 Query the INFORMATION_SCHEMA.VIEWS table

Usage example:

```
select * from information_schema.views;
```

You can view the relevant meta information of the view by querying this table, such as TABLE_CATALOG, TABLE_SCHEMA, TABLE_NAME, VIEW_DEFINITION, CHECK_OPTION, IS_UPDATABLE, DEFINER, SECURITY_TYPE, CHARACTER_SET_CLIENT, and COLLATION_CONNECTION .

```
+--+
→ -----
→
| TABLE_CATALOG | TABLE_SCHEMA | TABLE_NAME | VIEW_DEFINITION
→
→
→ | DEFINER | SECURITY_TYPE | CHARACTER_SET_CLIENT |
→ | COLLATION_CONNECTION |
```

```
+--+
| def      | test      | v          | SELECT `s`.`a` FROM `test`.`t` LEFT
| JOIN `test`.`s` ON `t`.`a`=`s`.`a` | CASCDED | NO | root@127.0.0.1
| DEFINER | utf8           | utf8_general_ci |
+--+
|                                     |
|                                     |
1 row in set (0.00 sec)
```

11.5.12.2.3 Use the HTTP APIs

Usage example:

```
curl http://127.0.0.1:10080/schema/test/v
```

By visiting `http://{TiDBIP}:10080/schema/{db}/{view}`, you can get all the metadata for the view.

```
{
  "id": 122,
  "name": {
    "O": "v",
    "L": "v"
  },
  "charset": "utf8",
  "collate": "utf8_general_ci",
  "cols": [
    {
      "id": 1,
      "name": {
        "O": "a",
        "L": "a"
      },
      "offset": 0,
      "origin_default": null,
      "default": null,
      "default_bit": null,
      "default_is_expr": false,
      "generated_expr_string": "",
      "generated_stored": false,
      "dependences": null,
      "type": {
        "Tp": 0,
```

```

    "Flag": 0,
    "Flen": 0,
    "Decimal": 0,
    "Charset": "",
    "Collate": "",
    "Elems": null
  },
  "state": 5,
  "comment": "",
  "hidden": false,
  "version": 0
}
],
"index_info": null,
"fk_info": null,
"state": 5,
"pk_is_handle": false,
"is_common_handle": false,
"comment": "",
"auto_inc_id": 0,
"auto_id_cache": 0,
"auto_rand_id": 0,
"max_col_id": 1,
"max_idx_id": 0,
"update_timestamp": 416801600091455490,
"ShardRowIDBits": 0,
"max_shard_row_id_bits": 0,
"auto_random_bits": 0,
"pre_split_regions": 0,
"partition": null,
"compression": "",
"view": {
  "view_algorithm": 0,
  "view_definer": {
    "Username": "root",
    "Hostname": "127.0.0.1",
    "CurrentUser": false,
    "AuthUsername": "root",
    "AuthHostname": "%"
  },
  "view_security": 0,
  "view_select": "SELECT `s`.`a` FROM `test`.`t` LEFT JOIN `test`.`s` ON `t` → `.` `a` = `s`.`a`",
  "view_checkoption": 1,
  "view_cols": null
}
]

```

```

},
"sequence": null,
"Lock": null,
"version": 3,
"tiflash_replica": null
}

```

11.5.12.3 Example

The following example creates a view, queries this view, and delete this view:

```
create table t(a int, b int);
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
insert into t values(1, 1),(2,2),(3,3);
```

```
Query OK, 3 rows affected (0.00 sec)
Records: 3 Duplicates: 0 Warnings: 0
```

```
create table s(a int);
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
insert into s values(2),(3);
```

```
Query OK, 2 rows affected (0.01 sec)
Records: 2 Duplicates: 0 Warnings: 0
```

```
create view v as select s.a from t left join s on t.a = s.a;
```

```
Query OK, 0 rows affected (0.01 sec)
```

```
select * from v;
```

```

+---+
| a |
+---+
| NULL |
|   2 |
|   3 |
+---+
3 rows in set (0.00 sec)

```

```
drop view v;
```

```
Query OK, 0 rows affected (0.02 sec)
```

11.5.12.4 Limitations

Currently, views in TiDB are subject to the following limitations:

- Materialized views are not supported yet.
- Views in TiDB are read-only and do not support write operations such as UPDATE, INSERT, DELETE, and TRUNCATE.
- For created views, the only supported DDL operation is DROP [VIEW | TABLE]

11.5.12.5 See also

- [CREATE VIEW](#)
- [DROP VIEW](#)

11.5.13 Partitioning

This document introduces TiDB's implementation of partitioning.

11.5.13.1 Partitioning types

This section introduces the types of partitioning in TiDB. Currently, TiDB supports [Range partitioning](#), [List partitioning](#), [List COLUMNS partitioning](#), and [Hash partitioning](#).

Range partitioning, List partitioning and List COLUMNS partitioning are used to resolve the performance issues caused by a large amount of deletions in the application, and support fast drop partition operations. Hash partitioning is used to scatter the data when there are a large amount of writes.

11.5.13.1.1 Range partitioning

When a table is partitioned by Range, each partition contains rows for which the partitioning expression value lies within a given Range. Ranges have to be contiguous but not overlapping. You can define it by using `VALUES LESS THAN`.

Assume you need to create a table that contains personnel records as follows:

```
CREATE TABLE employees (
    id INT NOT NULL,
    fname VARCHAR(30),
    lname VARCHAR(30),
    hired DATE NOT NULL DEFAULT '1970-01-01',
```

```

    separated DATE DEFAULT '9999-12-31',
    job_code INT,
    store_id INT NOT NULL
);

```

You can partition a table by Range in various ways as needed. For example, you can partition it by using the `store_id` column:

```

CREATE TABLE employees (
    id INT NOT NULL,
    fname VARCHAR(30),
    lname VARCHAR(30),
    hired DATE NOT NULL DEFAULT '1970-01-01',
    separated DATE DEFAULT '9999-12-31',
    job_code INT,
    store_id INT NOT NULL
)

PARTITION BY RANGE (store_id) (
    PARTITION p0 VALUES LESS THAN (6),
    PARTITION p1 VALUES LESS THAN (11),
    PARTITION p2 VALUES LESS THAN (16),
    PARTITION p3 VALUES LESS THAN (21)
);

```

In this partition scheme, all rows corresponding to employees whose `store_id` is 1 through 5 are stored in the `p0` partition while all employees whose `store_id` is 6 through 10 are stored in `p1`. Range partitioning requires the partitions to be ordered, from lowest to highest.

If you insert a row of data `(72, 'Tom', 'John', '2015-06-25', NULL, NULL, 15)`, it falls in the `p2` partition. But if you insert a record whose `store_id` is larger than 20, an error is reported because TiDB can not know which partition this record should be inserted into. In this case, you can use `MAXVALUE` when creating a table:

```

CREATE TABLE employees (
    id INT NOT NULL,
    fname VARCHAR(30),
    lname VARCHAR(30),
    hired DATE NOT NULL DEFAULT '1970-01-01',
    separated DATE DEFAULT '9999-12-31',
    job_code INT,
    store_id INT NOT NULL
)

PARTITION BY RANGE (store_id) (

```

```

    PARTITION p0 VALUES LESS THAN (6),
    PARTITION p1 VALUES LESS THAN (11),
    PARTITION p2 VALUES LESS THAN (16),
    PARTITION p3 VALUES LESS THAN MAXVALUE
);

```

MAXVALUE represents an integer value that is larger than all other integer values. Now, all records whose `store_id` is equal to or larger than 16 (the highest value defined) are stored in the p3 partition.

You can also partition a table by employees' job codes, which are the values of the `job_code` column. Assume that two-digit job codes stand for regular employees, three-digit codes stand for office and customer support personnel, and four-digit codes stand for managerial personnel. Then you can create a partitioned table like this:

```

CREATE TABLE employees (
    id INT NOT NULL,
    fname VARCHAR(30),
    lname VARCHAR(30),
    hired DATE NOT NULL DEFAULT '1970-01-01',
    separated DATE DEFAULT '9999-12-31',
    job_code INT,
    store_id INT NOT NULL
)

PARTITION BY RANGE (job_code) (
    PARTITION p0 VALUES LESS THAN (100),
    PARTITION p1 VALUES LESS THAN (1000),
    PARTITION p2 VALUES LESS THAN (10000)
);

```

In this example, all rows relating to regular employees are stored in the p0 partition, all office and customer support personnel in the p1 partition, and all managerial personnel in the p2 partition.

Besides splitting up the table by `store_id`, you can also partition a table by dates. For example, you can partition by employees' separation year:

```

CREATE TABLE employees (
    id INT NOT NULL,
    fname VARCHAR(30),
    lname VARCHAR(30),
    hired DATE NOT NULL DEFAULT '1970-01-01',
    separated DATE DEFAULT '9999-12-31',
    job_code INT,
    store_id INT
)

```

```
PARTITION BY RANGE ( YEAR(separated) ) (
    PARTITION p0 VALUES LESS THAN (1991),
    PARTITION p1 VALUES LESS THAN (1996),
    PARTITION p2 VALUES LESS THAN (2001),
    PARTITION p3 VALUES LESS THAN MAXVALUE
);
```

In Range partitioning, you can partition based on the values of the `timestamp` column and use the `unix_timestamp()` function, for example:

```
CREATE TABLE quarterly_report_status (
    report_id INT NOT NULL,
    report_status VARCHAR(20) NOT NULL,
    report_updated TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
        → CURRENT_TIMESTAMP
)

PARTITION BY RANGE ( UNIX_TIMESTAMP(report_updated) ) (
    PARTITION p0 VALUES LESS THAN ( UNIX_TIMESTAMP('2008-01-01 00:00:00') ),
    PARTITION p1 VALUES LESS THAN ( UNIX_TIMESTAMP('2008-04-01 00:00:00') ),
    PARTITION p2 VALUES LESS THAN ( UNIX_TIMESTAMP('2008-07-01 00:00:00') ),
    PARTITION p3 VALUES LESS THAN ( UNIX_TIMESTAMP('2008-10-01 00:00:00') ),
    PARTITION p4 VALUES LESS THAN ( UNIX_TIMESTAMP('2009-01-01 00:00:00') ),
    PARTITION p5 VALUES LESS THAN ( UNIX_TIMESTAMP('2009-04-01 00:00:00') ),
    PARTITION p6 VALUES LESS THAN ( UNIX_TIMESTAMP('2009-07-01 00:00:00') ),
    PARTITION p7 VALUES LESS THAN ( UNIX_TIMESTAMP('2009-10-01 00:00:00') ),
    PARTITION p8 VALUES LESS THAN ( UNIX_TIMESTAMP('2010-01-01 00:00:00') ),
    PARTITION p9 VALUES LESS THAN (MAXVALUE)
);
```

It is not allowed to use any other partitioning expression that contains the timestamp column.

Range partitioning is particularly useful when one or more of the following conditions are satisfied:

- You want to delete the old data. If you use the `employees` table in the previous example, you can delete all records of employees who left this company before the year 1991 by simply using `ALTER TABLE employees DROP PARTITION p0;`. It is faster than executing the `DELETE FROM employees WHERE YEAR(separated)<= 1990;` operation.
- You want to use a column that contains time or date values, or containing values arising from some other series.
- You need to frequently run queries on the columns used for partitioning. For example, when executing a query like `EXPLAIN SELECT COUNT(*)FROM employees WHERE → separated BETWEEN '2000-01-01' AND '2000-12-31' GROUP BY store_id;`,

TiDB can quickly know that only the data in the p2 partition needs to be scanned, because the other partitions do not match the `WHERE` condition.

11.5.13.1.2 List partitioning

Warning:

List partitioning is an experimental feature. It is not recommended that you use it in the production environment.

Before creating a List partitioned table, you need to set the value of the session variable `tidb_enable_list_partition` to `ON`.

```
set @@session.tidb_enable_list_partition = ON
```

Also, make sure that `tidb_enable_table_partition` is set to `ON`, which is the default setting.

List partitioning is similar to Range partitioning. Unlike Range partitioning, in List partitioning, the partitioning expression values for all rows in each partition are in a given value set. This value set defined for each partition can have any number of values but cannot have duplicate values. You can use the `PARTITION ... VALUES IN (...)` clause to define a value set.

Suppose that you want to create a personnel record table. You can create a table as follows:

```
CREATE TABLE employees (
    id INT NOT NULL,
    hired DATE NOT NULL DEFAULT '1970-01-01',
    store_id INT
);
```

Suppose that there are 20 stores distributed in 4 districts, as shown in the table below:

Region	Store ID Numbers
North	1, 2, 3, 4, 5
East	6, 7, 8, 9, 10
West	11, 12, 13, 14, 15
Central	16, 17, 18, 19, 20

If you want to store the personnel data of employees of the same region in the same partition, you can create a List partitioned table based on `store_id`:

```

CREATE TABLE employees (
    id INT NOT NULL,
    hired DATE NOT NULL DEFAULT '1970-01-01',
    store_id INT
)
PARTITION BY LIST (store_id) (
    PARTITION pNorth VALUES IN (1, 2, 3, 4, 5),
    PARTITION pEast VALUES IN (6, 7, 8, 9, 10),
    PARTITION pWest VALUES IN (11, 12, 13, 14, 15),
    PARTITION pCentral VALUES IN (16, 17, 18, 19, 20)
);

```

After creating the partitions as above, you can easily add or delete records related to a specific region in the table. For example, suppose that all stores in the East region (East) are sold to another company. Then all the row data related to the store employees of this region can be deleted by executing `ALTER TABLE employees TRUNCATE PARTITION pEast`, which is much more efficient than the equivalent statement `DELETE FROM employees WHERE → store_id IN (6, 7, 8, 9, 10)`.

You can also execute `ALTER TABLE employees DROP PARTITION pEast` to delete all related rows, but this statement also deletes the pEast partition from the table definition. In this situation, you must execute the `ALTER TABLE ... ADD PARTITION` statement to recover the original partitioning scheme of the table.

Unlike Range partitioning, List partitioning does not have a similar MAXVALUE partition to store all values that do not belong to other partitions. Instead, all expected values of the partition expression must be included in the `PARTITION ... VALUES IN (...)` clause. If the value to be inserted in an `INSERT` statement does not match the column value set of any partition, the statement fails to execute and an error is reported. See the following example:

```

test> CREATE TABLE t (
->   a INT,
->   b INT
-> )
-> PARTITION BY LIST (a) (
->   PARTITION p0 VALUES IN (1, 2, 3),
->   PARTITION p1 VALUES IN (4, 5, 6)
-> );
Query OK, 0 rows affected (0.11 sec)

test> INSERT INTO t VALUES (7, 7);
ERROR 1525 (HY000): Table has no partition for value 7

```

To ignore the error type above, you can use the `IGNORE` keyword. After using this keyword, if a row contains values that do not match the column value set of any partition, this row will not be inserted. Instead, any row with matched values is inserted, and no error

is reported:

```
test> TRUNCATE t;
Query OK, 1 row affected (0.00 sec)

test> INSERT IGNORE INTO t VALUES (1, 1), (7, 7), (8, 8), (3, 3), (5, 5);
Query OK, 3 rows affected, 2 warnings (0.01 sec)
Records: 5 Duplicates: 2 Warnings: 2

test> select * from t;
+---+---+
| a | b |
+---+---+
| 5 | 5 |
| 1 | 1 |
| 3 | 3 |
+---+---+
3 rows in set (0.01 sec)
```

11.5.13.1.3 List COLUMNS partitioning

Warning:

List COLUMNS partitioning is an experimental feature. It is not recommended that you use it in the production environment.

List COLUMNS partitioning is a variant of List partitioning. You can use multiple columns as partition keys. Besides the integer data type, you can also use the columns in the string, DATE, and DATETIME data types as partition columns.

Suppose that you want to divide the store employees from the following 12 cities into 4 regions, as shown in the following table:

Region	Cities
1	LosAngeles,Seattle, Houston
2	Chicago, Columbus, Boston
3	NewYork, LongIsland, Baltimore
4	Atlanta, Raleigh, Cincinnati

You can use List COLUMNS partitioning to create a table and store each row in the partition that corresponds to the employee's city, as shown below:

```

CREATE TABLE employees_1 (
    id INT NOT NULL,
    fname VARCHAR(30),
    lname VARCHAR(30),
    hired DATE NOT NULL DEFAULT '1970-01-01',
    separated DATE DEFAULT '9999-12-31',
    job_code INT,
    store_id INT,
    city VARCHAR(15)
)
PARTITION BY LIST COLUMNS(city) (
    PARTITION pRegion_1 VALUES IN('LosAngeles', 'Seattle', 'Houston'),
    PARTITION pRegion_2 VALUES IN('Chicago', 'Columbus', 'Boston'),
    PARTITION pRegion_3 VALUES IN('NewYork', 'LongIsland', 'Baltimore'),
    PARTITION pRegion_4 VALUES IN('Atlanta', 'Raleigh', 'Cincinnati')
);

```

Unlike List partitioning, in List COLUMNS partitioning, you do not need to use the expression in the `COLUMNS()` clause to convert column values to integers.

List COLUMNS partitioning can also be implemented using columns of the `DATE` and `DATETIME` types, as shown in the following example. This example uses the same names and columns as the previous `employees_1` table, but uses List COLUMNS partitioning based on the `hired` column:

```

CREATE TABLE employees_2 (
    id INT NOT NULL,
    fname VARCHAR(30),
    lname VARCHAR(30),
    hired DATE NOT NULL DEFAULT '1970-01-01',
    separated DATE DEFAULT '9999-12-31',
    job_code INT,
    store_id INT,
    city VARCHAR(15)
)
PARTITION BY LIST COLUMNS(hired) (
    PARTITION pWeek_1 VALUES IN('2020-02-01', '2020-02-02', '2020-02-03',
        '2020-02-04', '2020-02-05', '2020-02-06', '2020-02-07'),
    PARTITION pWeek_2 VALUES IN('2020-02-08', '2020-02-09', '2020-02-10',
        '2020-02-11', '2020-02-12', '2020-02-13', '2020-02-14'),
    PARTITION pWeek_3 VALUES IN('2020-02-15', '2020-02-16', '2020-02-17',
        '2020-02-18', '2020-02-19', '2020-02-20', '2020-02-21'),
    PARTITION pWeek_4 VALUES IN('2020-02-22', '2020-02-23', '2020-02-24',
        '2020-02-25', '2020-02-26', '2020-02-27', '2020-02-28')
);

```

In addition, you can also add multiple columns in the `COLUMNS()` clause. For example:

```
CREATE TABLE t (
    id int,
    name varchar(10)
)
PARTITION BY LIST COLUMNS(id,name) (
    partition p0 values IN ((1,'a'),(2,'b')),
    partition p1 values IN ((3,'c'),(4,'d')),
    partition p3 values IN ((5,'e'),(null,null))
);
```

11.5.13.1.4 Hash partitioning

Hash partitioning is used to make sure that data is evenly scattered into a certain number of partitions. With Range partitioning, you must specify the range of the column values for each partition when you use Range partitioning, while you just need to specify the number of partitions when you use Hash partitioning.

Partitioning by Hash requires you to append a `PARTITION BY HASH(expr)` clause to the `CREATE TABLE` statement. `expr` is an expression that returns an integer. It can be a column name if the type of this column is integer. In addition, you might also need to append `PARTITIONS num`, where `num` is a positive integer indicating how many partitions a table is divided into.

The following operation creates a Hash partitioned table, which is divided into 4 partitions by `store_id`:

```
CREATE TABLE employees (
    id INT NOT NULL,
    fname VARCHAR(30),
    lname VARCHAR(30),
    hired DATE NOT NULL DEFAULT '1970-01-01',
    separated DATE DEFAULT '9999-12-31',
    job_code INT,
    store_id INT
)

PARTITION BY HASH(store_id)
PARTITIONS 4;
```

If `PARTITIONS num` is not specified, the default number of partitions is 1.

You can also use an SQL expression that returns an integer for `expr`. For example, you can partition a table by the hire year:

```
CREATE TABLE employees (
    id INT NOT NULL,
```

```

        fname VARCHAR(30),
        lname VARCHAR(30),
        hired DATE NOT NULL DEFAULT '1970-01-01',
        separated DATE DEFAULT '9999-12-31',
        job_code INT,
        store_id INT
    )

PARTITION BY HASH( YEAR(hired) )
PARTITIONS 4;

```

The most efficient Hash function is one which operates upon a single table column, and whose value increases or decreases consistently with the column value.

For example, `date_col` is a column whose type is `DATE`, and the value of the `TO_DAYS` \rightarrow (`date_col`) expression varies with the value of `date_col`. `YEAR(date_col)` is different from `TO_DAYS(date_col)`, because not every possible change in `date_col` produces an equivalent change in `YEAR(date_col)`.

In contrast, assume that you have an `int_col` column whose type is `INT`. Now consider about the expression `POW(5-int_col,3)+ 6`. It is not a good Hash function though, because as the value of `int_col` changes, the result of the expression does not change proportionally. A value change in `int_col` might result in a huge change in the expression result. For example, when `int_col` changes from 5 to 6, the change of the expression result is -1. But the result change might be -7 when `int_col` changes from 6 to 7.

In conclusion, when the expression has a form that is closer to $y = cx$, it is more suitable to be a Hash function. Because the more non-linear an expression is, the more unevenly scattered the data among the partitions tends to be.

In theory, pruning is also possible for expressions involving more than one column value, but determining which of such expressions are suitable can be quite difficult and time-consuming. For this reason, the use of hashing expressions involving multiple columns is not particularly recommended.

When using `PARTITION BY HASH`, TiDB decides which partition the data should fall into based on the modulus of the result of the expression. In other words, if a partitioning expression is `expr` and the number of partitions is `num`, `MOD(expr, num)` decides the partition in which the data is stored. Assume that `t1` is defined as follows:

```

CREATE TABLE t1 (col1 INT, col2 CHAR(5), col3 DATE)
PARTITION BY HASH( YEAR(col3) )
PARTITIONS 4;

```

When you insert a row of data into `t1` and the value of `col3` is '2005-09-15', then this row is inserted into partition 1:

```

MOD(YEAR('2005-09-01'),4)
= MOD(2005,4)

```

```
= 1
```

11.5.13.1.5 How TiDB partitioning handles NULL

It is allowed in TiDB to use NULL as the calculation result of a partitioning expression.

Note:

NULL is not an integer. TiDB's partitioning implementation treats NULL as being less than any other integer values, just as ORDER BY does.

Handling of NULL with Range partitioning

When you insert a row into a table partitioned by Range, and the column value used to determine the partition is NULL, then this row is inserted into the lowest partition.

```
CREATE TABLE t1 (
    c1 INT,
    c2 VARCHAR(20)
)

PARTITION BY RANGE(c1) (
    PARTITION p0 VALUES LESS THAN (0),
    PARTITION p1 VALUES LESS THAN (10),
    PARTITION p2 VALUES LESS THAN MAXVALUE
);
```

```
Query OK, 0 rows affected (0.09 sec)
```

```
select * from t1 partition(p0);
```

```
+----+-----+
| c1 | c2   |
+----+-----+
| NULL | mothra |
+----+-----+
1 row in set (0.00 sec)
```

```
select * from t1 partition(p1);
```

```
Empty set (0.00 sec)
```

```
select * from t1 partition(p2);
```

```
Empty set (0.00 sec)
```

Drop the p0 partition and verify the result:

```
alter table t1 drop partition p0;
```

```
Query OK, 0 rows affected (0.08 sec)
```

```
select * from t1;
```

```
Empty set (0.00 sec)
```

Handling of NULL with Hash partitioning

When partitioning tables by Hash, there is a different way of handling NULL value - if the calculation result of the partitioning expression is NULL, it is considered as 0.

```
CREATE TABLE th (
    c1 INT,
    c2 VARCHAR(20)
)
```

```
PARTITION BY HASH(c1)
PARTITIONS 2;
```

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

```
INSERT INTO th VALUES (NULL, 'mothra'), (0, 'gigan');
```

```
Query OK, 2 rows affected (0.04 sec)
```

```
select * from th partition (p0);
```

c1	c2
NULL	mothra
0	gigan

```
2 rows in set (0.00 sec)
```

```
select * from th partition (p1);
```

```
Empty set (0.00 sec)
```

You can see that the inserted record (NULL, 'mothra') falls into the same partition as (0, 'gigan').

Note: NULL values by Hash partitions in TiDB are handled in the same way as described in [How MySQL Partitioning Handles NULL](#), which, however, is not consistent with the actual behavior of MySQL. In other words, MySQL's implementation in this case is not consistent with its documentation.

In this case, the actual behavior of TiDB is in line with the description of this document.

11.5.13.2 Partition management

For LIST and RANGE partitioned tables, you can add and drop partitions using the `ALTER TABLE <table name> ADD PARTITION (<partition specification>)` or `ALTER TABLE <table name> DROP PARTITION <list of partitions>` statement.

For LIST and RANGE partitioned tables, `REORGANIZE PARTITION` is not yet supported.

For HASH partitioned tables, `COALESCE PARTITION` and `ADD PARTITION` are not yet supported.

`EXCHANGE PARTITION` works by swapping a partition and a non-partitioned table, similar to how renaming a table like `RENAME TABLE t1 TO t1_tmp, t2 TO t1, t1_tmp TO t2` works.

For example, `ALTER TABLE partitioned_table EXCHANGE PARTITION p1 WITH TABLE non_partitioned_table` swaps the `non_partitioned_table` table in the `p1` partition with the `partitioned_table` table.

Ensure that all rows that you are exchanging into the partition match the partition definition; otherwise, these rows will not be found and cause unexpected issues.

Warning:

`EXCHANGE PARTITION` is an experimental feature. It is not recommended to use it in a production environment. To enable it, set the `tidb_enable_exchange_partition` system variable to `ON`.

11.5.13.2.1 Range partition management

Create a partitioned table:

```
CREATE TABLE members (
    id INT,
    fname VARCHAR(25),
    lname VARCHAR(25),
    dob DATE
)

PARTITION BY RANGE( YEAR(dob) ) (
    PARTITION p0 VALUES LESS THAN (1980),
    PARTITION p1 VALUES LESS THAN (1990),
    PARTITION p2 VALUES LESS THAN (2000)
);
```

Drop a partition:

```
ALTER TABLE members DROP PARTITION p2;
```

```
Query OK, 0 rows affected (0.03 sec)
```

Empty a partition:

```
ALTER TABLE members TRUNCATE PARTITION p1;
```

```
Query OK, 0 rows affected (0.03 sec)
```

Note:

`ALTER TABLE ... REORGANIZE PARTITION` is currently unsupported in TiDB.

Add a partition:

```
ALTER TABLE members ADD PARTITION (PARTITION p3 VALUES LESS THAN (2010));
```

When partitioning tables by Range, `ADD PARTITION` can be only appended to the very end of a partition list. If it is appended to an existing Range partition, an error is reported:

```
ALTER TABLE members
    ADD PARTITION (
        PARTITION n VALUES LESS THAN (1970));
```

```
ERROR 1463 (HY000): VALUES LESS THAN value must be strictly »
increasing for each partition
```

11.5.13.2.2 Hash partition management

Unlike Range partitioning, `DROP PARTITION` is not supported in Hash partitioning.

Currently, `ALTER TABLE ... COALESCE PARTITION` is not supported in TiDB as well. For partition management statements that are not currently supported, TiDB returns an error.

```
alter table members optimize partition p0;
```

```
ERROR 8200 (HY000): Unsupported optimize partition
```

11.5.13.3 Partition pruning

Partition pruning is an optimization which is based on a very simple idea - do not scan the partitions that do not match.

Assume that you create a partitioned table `t1`:

```
CREATE TABLE t1 (
    fname VARCHAR(50) NOT NULL,
    lname VARCHAR(50) NOT NULL,
    region_code TINYINT UNSIGNED NOT NULL,
    dob DATE NOT NULL
)

PARTITION BY RANGE( region_code ) (
    PARTITION p0 VALUES LESS THAN (64),
    PARTITION p1 VALUES LESS THAN (128),
    PARTITION p2 VALUES LESS THAN (192),
    PARTITION p3 VALUES LESS THAN MAXVALUE
);
```

If you want to get the result of this `SELECT` statement:

```
SELECT fname, lname, region_code, dob
  FROM t1
 WHERE region_code > 125 AND region_code < 130;
```

It is evident that the result falls in either the `p1` or the `p2` partition, that is, you just need to search for the matching rows in `p1` and `p2`. Excluding the unneeded partitions is so-called “pruning”. If the optimizer is able to prune a part of partitions, the execution of the query in the partitioned table will be much faster than that in a non-partitioned table.

The optimizer can prune partitions through WHERE conditions in the following two scenarios:

- partition_column = constant
- partition_column IN (constant1, constant2, ..., constantN)

11.5.13.3.1 Some cases for partition pruning to take effect

1. Partition pruning uses the query conditions on the partitioned table, so if the query conditions can not be pushed down to the partitioned table according to the planner's optimization rules, partition pruning does not apply for this query.

For example:

```
create table t1 (x int) partition by range (x) (
    partition p0 values less than (5),
    partition p1 values less than (10));
create table t2 (x int);
```

```
explain select * from t1 left join t2 on t1.x = t2.x where t2.x > 5;
```

In this query, the left out join is converted to the inner join, and then `t1.x > 5` is derived from `t1.x = t2.x` and `t2.x > 5`, so it could be used in partition pruning and only the partition `p1` remains.

```
explain select * from t1 left join t2 on t1.x = t2.x and t2.x > 5;
```

In this query, `t2.x > 5` can not be pushed down to the `t1` partitioned table, so partition pruning would not take effect for this query.

2. Since partition pruning is done during the plan optimizing phase, it does not apply for those cases that filter conditions are unknown until the execution phase.

For example:

```
create table t1 (x int) partition by range (x) (
    partition p0 values less than (5),
    partition p1 values less than (10));
```

```
explain select * from t2 where x < (select * from t1 where t2.x < t1.x
    ↳ and t2.x < 2);
```

This query reads a row from `t2` and uses the result for the subquery on `t1`. Theoretically, partition pruning could benefit from `t1.x > val` expression in the subquery, but it does not take effect there as that happens in the execution phase.

3. As a result of a limitation from current implementation, if a query condition can not be pushed down to TiKV, it can not be used by the partition pruning.

Take the `fn(col)` expression as an example. If the TiKV coprocessor supports this `fn` function, `fn(col)` may be pushed down to the leaf node (that is, partitioned table) according to the predicate push-down rule during the plan optimizing phase, and partition pruning can use it.

If the TiKV coprocessor does not support this `fn` function, `fn(col)` would not be pushed down to the leaf node. Instead, it becomes a `Selection` node above the leaf node. The current partition pruning implementation does not support this kind of plan tree.

4. For Hash partition, the only query supported by partition pruning is the equal condition.
5. For Range partition, for partition pruning to take effect, the partition expression must be in those forms: `col` or `fn(col)`, and the query condition must be one of `>`, `<`, `=`, `>=`, and `<=`. If the partition expression is in the form of `fn(col)`, the `fn` function must be monotonous.

If the `fn` function is monotonous, for any `x` and `y`, if `x > y`, then `fn(x) > fn(y)`. Then this `fn` function can be called strictly monotonous. For any `x` and `y`, if `x > y`, then `fn(x) >= fn(y)`. In this case, `fn` could also be called “monotonous”. In theory, all monotonous functions are supported by partition pruning.

Currently, partition pruning in TiDB only support those monotonous functions:

```
unix_timestamp
to_days
```

For example, the partition expression is a simple column:

```
create table t (id int) partition by range (id) (
    partition p0 values less than (5),
    partition p1 values less than (10));
select * from t where t > 6;
```

Or the partition expression is in the form of `fn(col)` where `fn` is `to_days`:

```
create table t (dt datetime) partition by range (to_days(id)) (
    partition p0 values less than (to_days('2020-04-01')),
    partition p1 values less than (to_days('2020-05-01')));
select * from t where t > '2020-04-18';
```

An exception is `floor(unix_timestamp())` as the partition expression. TiDB does some optimization for that case by case, so it is supported by partition pruning.

```
create table t (ts timestamp(3) not null default current_timestamp(3))
partition by range (floor(unix_timestamp(ts))) (
```

```

partition p0 values less than (unix_timestamp('2020-04-01
    ↪ 00:00:00')),
partition p1 values less than (unix_timestamp('2020-05-01
    ↪ 00:00:00'));
select * from t where t > '2020-04-18 02:00:42.123';

```

11.5.13.4 Partition selection

SELECT statements support partition selection, which is implemented by using a PARTITION option.

```

CREATE TABLE employees (
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    fname VARCHAR(25) NOT NULL,
    lname VARCHAR(25) NOT NULL,
    store_id INT NOT NULL,
    department_id INT NOT NULL
)

PARTITION BY RANGE(id) (
    PARTITION p0 VALUES LESS THAN (5),
    PARTITION p1 VALUES LESS THAN (10),
    PARTITION p2 VALUES LESS THAN (15),
    PARTITION p3 VALUES LESS THAN MAXVALUE
);

INSERT INTO employees VALUES
('Bob', 'Taylor', 3, 2), ('Frank', 'Williams', 1, 2),
('Ellen', 'Johnson', 3, 4), ('Jim', 'Smith', 2, 4),
('Mary', 'Jones', 1, 1), ('Linda', 'Black', 2, 3),
('Ed', 'Jones', 2, 1), ('June', 'Wilson', 3, 1),
('Andy', 'Smith', 1, 3), ('Lou', 'Waters', 2, 4),
('Jill', 'Stone', 1, 4), ('Roger', 'White', 3, 2),
('Howard', 'Andrews', 1, 2), ('Fred', 'Goldberg', 3, 3),
('Barbara', 'Brown', 2, 3), ('Alice', 'Rogers', 2, 2),
('Mark', 'Morgan', 3, 3), ('Karen', 'Cole', 3, 2);

```

You can view the rows stored in the p1 partition:

```
SELECT * FROM employees PARTITION (p1);
```

id	fname	lname	store_id	department_id
5	Mary	Jones	1	1

```

| 6 | Linda | Black |      2 |      3 |
| 7 | Ed    | Jones |      2 |      1 |
| 8 | June | Wilson|      3 |      1 |
| 9 | Andy | Smith|      1 |      3 |
+---+-----+-----+-----+
5 rows in set (0.00 sec)

```

If you want to get the rows in multiple partitions, you can use a list of partition names which are separated by commas. For example, `SELECT * FROM employees PARTITION (p1, p2)` returns all rows in the p1 and p2 partitions.

When you use partition selection, you can still use `WHERE` conditions and options such as `ORDER BY` and `LIMIT`. It is also supported to use aggregation options such as `HAVING` and `GROUP BY`.

```
SELECT * FROM employees PARTITION (p0, p2)
  WHERE lname LIKE 'S%';
```

```

+---+-----+-----+-----+-----+
| id | fname | lname | store_id | department_id |
+---+-----+-----+-----+
| 4 | Jim   | Smith |      2 |      4 |
| 11 | Jill  | Stone |      1 |      4 |
+---+-----+-----+-----+
2 rows in set (0.00 sec)

```

```
SELECT id, CONCAT(fname, ' ', lname) AS name
  FROM employees PARTITION (p0) ORDER BY lname;
```

```

+---+-----+
| id | name      |
+---+-----+
| 3 | Ellen Johnson |
| 4 | Jim Smith   |
| 1 | Bob Taylor   |
| 2 | Frank Williams |
+---+-----+
4 rows in set (0.06 sec)

```

```
SELECT store_id, COUNT(department_id) AS c
  FROM employees PARTITION (p1,p2,p3)
 GROUP BY store_id HAVING c > 4;
```

```

+---+-----+
| c | store_id |
+---+-----+

```

```
+---+-----+
| 5 |      2 |
| 5 |      3 |
+---+-----+
2 rows in set (0.00 sec)
```

Partition selection is supported for all types of table partitioning, including Range partitioning and Hash partitioning. For Hash partitions, if partition names are not specified, p0, p1, p2,..., or pN-1 is automatically used as the partition name.

SELECT in INSERT ... SELECT can also use partition selection.

11.5.13.5 Restrictions and limitations on partitions

This section introduces some restrictions and limitations on partitioned tables in TiDB.

11.5.13.5.1 Partitioning keys, primary keys and unique keys

This section discusses the relationship of partitioning keys with primary keys and unique keys. The rule governing this relationship can be expressed as follows: **Every unique key on the table must use every column in the table's partitioning expression.** This also includes the table's primary key, because it is by definition a unique key.

For example, the following table creation statements are invalid:

```
CREATE TABLE t1 (
    col1 INT NOT NULL,
    col2 DATE NOT NULL,
    col3 INT NOT NULL,
    col4 INT NOT NULL,
    UNIQUE KEY (col1, col2)
)

PARTITION BY HASH(col3)
PARTITIONS 4;

CREATE TABLE t2 (
    col1 INT NOT NULL,
    col2 DATE NOT NULL,
    col3 INT NOT NULL,
    col4 INT NOT NULL,
    UNIQUE KEY (col1),
    UNIQUE KEY (col3)
)

PARTITION BY HASH(col1 + col3)
PARTITIONS 4;
```

In each case, the proposed table has at least one unique key that does not include all columns used in the partitioning expression.

The valid statements are as follows:

```
CREATE TABLE t1 (
    col1 INT NOT NULL,
    col2 DATE NOT NULL,
    col3 INT NOT NULL,
    col4 INT NOT NULL,
    UNIQUE KEY (col1, col2, col3)
)

PARTITION BY HASH(col3)
PARTITIONS 4;

CREATE TABLE t2 (
    col1 INT NOT NULL,
    col2 DATE NOT NULL,
    col3 INT NOT NULL,
    col4 INT NOT NULL,
    UNIQUE KEY (col1, col3)
)

PARTITION BY HASH(col1 + col3)
PARTITIONS 4;
```

The following example displays an error:

```
CREATE TABLE t3 (
    col1 INT NOT NULL,
    col2 DATE NOT NULL,
    col3 INT NOT NULL,
    col4 INT NOT NULL,
    UNIQUE KEY (col1, col2),
    UNIQUE KEY (col3)
)

PARTITION BY HASH(col1 + col3)
PARTITIONS 4;
```

ERROR 1491 (HY000): A PRIMARY KEY must include all columns in the table's
 ↪ partitioning function

The CREATE TABLE statement fails because both `col1` and `col3` are included in the proposed partitioning key, but neither of these columns is part of both of unique keys on the table. After the following modifications, the CREATE TABLE statement becomes valid:

```

CREATE TABLE t3 (
    col1 INT NOT NULL,
    col2 DATE NOT NULL,
    col3 INT NOT NULL,
    col4 INT NOT NULL,
    UNIQUE KEY (col1, col2, col3),
    UNIQUE KEY (col1, col3)
)
PARTITION BY HASH(col1 + col3)
    PARTITIONS 4;

```

The following table cannot be partitioned at all, because there is no way to include in a partitioning key any columns that belong to both unique keys:

```

CREATE TABLE t4 (
    col1 INT NOT NULL,
    col2 INT NOT NULL,
    col3 INT NOT NULL,
    col4 INT NOT NULL,
    UNIQUE KEY (col1, col3),
    UNIQUE KEY (col2, col4)
);

```

Because every primary key is by definition a unique key, so the next two statements are invalid:

```

CREATE TABLE t5 (
    col1 INT NOT NULL,
    col2 DATE NOT NULL,
    col3 INT NOT NULL,
    col4 INT NOT NULL,
    PRIMARY KEY(col1, col2)
)

PARTITION BY HASH(col3)
    PARTITIONS 4;

CREATE TABLE t6 (
    col1 INT NOT NULL,
    col2 DATE NOT NULL,
    col3 INT NOT NULL,
    col4 INT NOT NULL,
    PRIMARY KEY(col1, col3),
    UNIQUE KEY(col2)
)

```

```
PARTITION BY HASH( YEAR(col2) )
PARTITIONS 4;
```

In the above examples, the primary key does not include all columns referenced in the partitioning expression. After adding the missing column in the primary key, the CREATE TABLE statement becomes valid:

```
CREATE TABLE t5 (
    col1 INT NOT NULL,
    col2 DATE NOT NULL,
    col3 INT NOT NULL,
    col4 INT NOT NULL,
    PRIMARY KEY(col1, col2, col3)
)
PARTITION BY HASH(col3)
PARTITIONS 4;
CREATE TABLE t6 (
    col1 INT NOT NULL,
    col2 DATE NOT NULL,
    col3 INT NOT NULL,
    col4 INT NOT NULL,
    PRIMARY KEY(col1, col2, col3),
    UNIQUE KEY(col2)
)
PARTITION BY HASH( YEAR(col2) )
PARTITIONS 4;
```

If a table has neither unique keys nor primary keys, then this restriction does not apply.

When you change tables using DDL statements, you also need to consider this restriction when adding a unique index. For example, when you create a partitioned table as shown below:

```
CREATE TABLE t_no_pk (c1 INT, c2 INT)
PARTITION BY RANGE(c1) (
    PARTITION p0 VALUES LESS THAN (10),
    PARTITION p1 VALUES LESS THAN (20),
    PARTITION p2 VALUES LESS THAN (30),
    PARTITION p3 VALUES LESS THAN (40)
);
```

```
Query OK, 0 rows affected (0.12 sec)
```

You can add a non-unique index by using ALTER TABLE statements. But if you want to add a unique index, the c1 column must be included in the unique index.

When using a partitioned table, you cannot specify the prefix index as a unique attribute:

```
CREATE TABLE t (a varchar(20), b blob,
    UNIQUE INDEX (a(5)))
PARTITION by range columns (a) (
    PARTITION p0 values less than ('aaaaa'),
    PARTITION p1 values less than ('bbbb'),
    PARTITION p2 values less than ('ccccc'));
```

```
ERROR 1503 (HY000): A UNIQUE INDEX must include all columns in the table's
↪ partitioning function
```

11.5.13.5.2 Partitioning limitations relating to functions

Only the functions shown in the following list are allowed in partitioning expressions:

```
ABS()
CEILING()
DATEDIFF()
DAY()
DAYOFMONTH()
DAYOFWEEK()
DAYOFYEAR()
EXTRACT() (see EXTRACT() function with WEEK specifier)
FLOOR()
HOUR()
MICROSECOND()
MINUTE()
MOD()
MONTH()
QUARTER()
SECOND()
TIME_TO_SEC()
TO_DAYS()
TO_SECONDS()
UNIX_TIMESTAMP() (with TIMESTAMP columns)
WEEKDAY()
YEAR()
YEARWEEK()
```

11.5.13.5.3 Compatibility with MySQL

Currently, TiDB supports Range partitioning, List partitioning, List COLUMNS partitioning, and Hash partitioning. Other partitioning types that are available in MySQL such as key partitioning are not supported yet in TiDB.

For a table partitioned by RANGE COLUMNS, currently TiDB only supports using a single partitioning column.

With regard to partition management, any operation that requires moving data in the bottom implementation is not supported currently, including but not limited to: adjust the number of partitions in a Hash partitioned table, modify the Range of a Range partitioned table, merge partitions and exchange partitions.

For the unsupported partitioning types, when you create a table in TiDB, the partitioning information is ignored and the table is created in the regular form with a warning reported.

The LOAD DATA syntax does not support partition selection currently in TiDB.

```
create table t (id int, val int) partition by hash(id) partitions 4;
```

The regular LOAD DATA operation is supported:

```
load local data infile "xxx" into t ...
```

But Load Data does not support partition selection:

```
load local data infile "xxx" into t partition (p1)...
```

For a partitioned table, the result returned by `select * from t` is unordered between the partitions. This is different from the result in MySQL, which is ordered between the partitions but unordered inside the partitions.

```
create table t (id int, val int) partition by range (id) (
    partition p0 values less than (3),
    partition p1 values less than (7),
    partition p2 values less than (11));
```

```
Query OK, 0 rows affected (0.10 sec)
```

```
insert into t values (1, 2), (3, 4),(5, 6),(7,8),(9,10);
```

```
Query OK, 5 rows affected (0.01 sec)
Records: 5 Duplicates: 0 Warnings: 0
```

TiDB returns a different result every time, for example:

```
select * from t;
```

id	val
7	8
9	10
1	2

```
|   3 |   4 |
|   5 |   6 |
+----+----+
5 rows in set (0.00 sec)
```

The result returned in MySQL:

```
select * from t;
```

```
+----+----+
| id | val |
+----+----+
|   1 |   2 |
|   3 |   4 |
|   5 |   6 |
|   7 |   8 |
|   9 |  10 |
+----+----+
5 rows in set (0.00 sec)
```

The `tidb_enable_list_partition` environment variable controls whether to enable the partitioned table feature. If this variable is set to `OFF`, the partition information will be ignored when a table is created, and this table will be created as a normal table.

This variable is only used in table creation. After the table is created, modify this variable value takes no effect. For details, see [system variables](#).

11.5.13.5.4 Dynamic pruning mode

Warning:

This is still an experimental feature. It is **NOT** recommended that you use it in the production environment.

TiDB accesses partitioned tables in one of the two modes: `dynamic` mode and `static` mode. Currently, `static` mode is used by default. If you want to enable `dynamic` mode, you need to manually set the `tidb_partition_prune_mode` variable to `dynamic`.

```
set @@session.tidb_partition_prune_mode = 'dynamic'
```

In `static` mode, TiDB accesses each partition separately using multiple operators, and then merges the results using `Union`. The following example is a simple read operation where TiDB merges the results of two corresponding partitions using `Union`:

```

mysql> create table t1(id int, age int, key(id)) partition by range(id) (
    ->     partition p0 values less than (100),
    ->     partition p1 values less than (200),
    ->     partition p2 values less than (300),
    ->     partition p3 values less than (400));
Query OK, 0 rows affected (0.01 sec)

mysql> explain select * from t1 where id < 150;
+---+
   → -----+-----+-----+
   →
   →
| id           | estRows | task      | access object      |
| operator info          |          |
+---+
   → -----+-----+-----+
   →
   →
| PartitionUnion_9       | 6646.67 | root      |                   |
   →
   →
| -TableReader_12        | 3323.33 | root      |                   | data
   → :Selection_11          |
| -Selection_11          | 3323.33 | cop[tikv] |                   | lt(
   → test.t1.id, 150)          |
| -TableFullScan_10       | 10000.00 | cop[tikv] | table:t1, partition:p0
   → | keep order:false, stats:pseudo |
| -TableReader_18        | 3323.33 | root      |                   | data
   → :Selection_17          |
| -Selection_17          | 3323.33 | cop[tikv] |                   | lt(
   → test.t1.id, 150)          |
| -TableFullScan_16       | 10000.00 | cop[tikv] | table:t1, partition:p1
   → | keep order:false, stats:pseudo |
+---+
   → -----+-----+-----+
   →
7 rows in set (0.00 sec)

```

In dynamic mode, each operator supports direct access to multiple partitions, so TiDB no longer uses Union.

```
mysql> set @@session.tidb_partition_prune_mode = 'dynamic';
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> explain select * from t1 where id < 150;
```

```

+----+-----+-----+-----+-----+
| id | estRows | task      | access object | operator info |
+----+-----+-----+-----+-----+
| TableReader_7       | 3323.33 | root      | partition:p0,p1 | data:          |
|   ↗ Selection_6     |           |           |               |               |
| -Selection_6        | 3323.33 | cop[tikv] |               | lt(test.t1.id, |
|   ↗ 150              |           |           |               |               |
|   -TableFullScan_5  | 10000.00 | cop[tikv] | table:t1    | keep order:   |
|     ↗ false, stats:pseudo |           |           |               |               |
+----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

```

From the above query results, you can see that the `Union` operator in the execution plan disappears while the partition pruning still takes effect and the execution plan only accesses `p0` and `p1`.

`dynamic` mode makes execution plans simpler and clearer. Omitting the `Union` operation can improve the execution efficiency and avoid the problem of `Union` concurrent execution. In addition, `dynamic` mode also solves two problems that cannot be solved in `static` mode:

- Plan Cache cannot be used. (See example 1 and 2)
- Execution plans with `IndexJoin` cannot be used. (See example 3 and 4)

Example 1: In the following example, the Plan Cache feature is enabled in the configuration file and the same query is executed twice in `static` mode:

```

mysql> set @a=150;
Query OK, 0 rows affected (0.00 sec)

mysql> set @@tidb_partition_prune_mode = 'static';
Query OK, 0 rows affected (0.00 sec)

mysql> prepare stmt from 'select * from t1 where id < ?';
Query OK, 0 rows affected (0.00 sec)

mysql> execute stmt using @a;
Empty set (0.00 sec)

mysql> execute stmt using @a;
Empty set (0.00 sec)

```

```
-- In static mode, when the same query is executed twice, the cache cannot
-- be hit at the second time.
mysql> select @@last_plan_from_cache;
+-----+
| @@last_plan_from_cache |
+-----+
| 0 |
+-----+
1 row in set (0.00 sec)
```

The `last_plan_from_cache` variable can show whether the last query hits the Plan Cache or not. From example 1, you can see that in `static` mode, even if the same query is executed multiple times on the partitioned table, the Plan Cache is not hit.

Example 2: In the following example, the same operations are performed in `dynamic` mode as done in example 1:

```
mysql> set @@tidb_partition_prune_mode = 'dynamic';
Query OK, 0 rows affected (0.00 sec)

mysql> prepare stmt from 'select * from t1 where id < ?';
Query OK, 0 rows affected (0.00 sec)

mysql> execute stmt using @a;
Empty set (0.00 sec)

mysql> execute stmt using @a;
Empty set (0.00 sec)

-- In dynamic mode, the cache can be hit at the second time.
mysql> select @@last_plan_from_cache;
+-----+
| @@last_plan_from_cache |
+-----+
| 1 |
+-----+
1 row in set (0.00 sec)
```

From example 2, you can see that in `dynamic` mode, querying the partitioned table hits the Plan Cache.

Example 3: In the following example, a query is performed in `static` mode using the execution plan with `IndexJoin`:

```
mysql> create table t2(id int, code int);
Query OK, 0 rows affected (0.01 sec)
```

```

mysql> set @@tidb_partition_prune_mode = 'static';
Query OK, 0 rows affected (0.00 sec)

mysql> explain select /*+ TIDB_INLJ(t1, t2) */ t1.* from t1, t2 where t2.
      ↪ code = 0 and t2.id = t1.id;
+---+
| id           | estRows | task      | access object      |
| operator info |          |           |                   |
+---+
| HashJoin_13   | 12.49   | root      |                   |
|   ↪ inner join, equal:[eq(test.t1.id, test.t2.id)] |
|   -TableReader_42(Build) | 9.99   | root      |                   |
|     ↪ data:Selection_41           |
|       -Selection_41    | 9.99   | cop[tikv] |                   |
|         ↪ (test.t2.code, 0), not(isnull(test.t2.id)) |
|           -TableFullScan_40 | 10000.00 | cop[tikv] | table:t2        |
|             ↪ keep order:false, stats:pseudo           |
|   -PartitionUnion_15(Probe) | 39960.00 | root      |                   |
|     ↪
|       -TableReader_18    | 9990.00 | root      |                   |
|         ↪ data:Selection_17           |
|           -Selection_17    | 9990.00 | cop[tikv] |                   |
|             ↪ not(isnull(test.t1.id))           |
|               -TableFullScan_16 | 10000.00 | cop[tikv] | table:t1, partition:
|                 ↪ p0 | keep order:false, stats:pseudo           |
|   -TableReader_24        | 9990.00 | root      |                   |
|     ↪ data:Selection_23           |
|       -Selection_23    | 9990.00 | cop[tikv] |                   |
|         ↪ not(isnull(test.t1.id))           |
|           -TableFullScan_22 | 10000.00 | cop[tikv] | table:t1, partition:
|             ↪ p1 | keep order:false, stats:pseudo           |
|   -TableReader_30        | 9990.00 | root      |                   |
|     ↪ data:Selection_29           |
|       -Selection_29    | 9990.00 | cop[tikv] |                   |
|         ↪ not(isnull(test.t1.id))           |
|           -TableFullScan_28 | 10000.00 | cop[tikv] | table:t1, partition:
|             ↪ p2 | keep order:false, stats:pseudo           |
|   -TableReader_36        | 9990.00 | root      |                   |
|     ↪ data:Selection_35           |
|       -Selection_35    | 9990.00 | cop[tikv] |                   |
|         ↪ not(isnull(test.t1.id))           |

```

```

|   -TableFullScan_34    | 10000.00 | cop[tikv] | table:t1, partition:
|   ↳ p3 | keep order:false, stats:pseudo |
+--+
|   ↳ -----
|   ↳
17 rows in set, 1 warning (0.00 sec)

```

From example 3, you can see that even if the TIDB_INLJ hint is used, the query on the partitioned table cannot select the execution plan with IndexJoin.

Example 4: In the following example, the query is performed in dynamic mode using the execution plan with IndexJoin:

```

mysql> set @@tidb_partition_prune_mode = 'dynamic';
Query OK, 0 rows affected (0.00 sec)

mysql> explain select /*+ TIDB_INLJ(t1, t2) */ t1.* from t1, t2 where t2.
      ↳ code = 0 and t2.id = t1.id;
+--+
| id          | estRows | task      | access object      |
| operator info
| ↳
| ↳ |
+--+
| IndexJoin_11 | 12.49   | root      |                   |
|   ↳ inner join, inner:IndexLookUp_10, outer key:test.t2.id, inner key:
|   ↳ test.t1.id, equal cond:eq(test.t2.id, test.t1.id) |
|   -TableReader_16(Build) | 9.99    | root      |                   |
|   ↳ data:Selection_15
|   ↳
|   ↳ |
|   -Selection_15 | 9.99    | cop[tikv] |                   |
|   ↳ eq(test.t2.code, 0), not(isnull(test.t2.id))
|   ↳
|   -TableFullScan_14 | 10000.00 | cop[tikv] | table:t2      |
|   ↳ keep order:false, stats:pseudo
|   ↳
|   ↳ |
|   -IndexLookUp_10(Probe) | 1.25    | root      | partition:all |
|   ↳
|   ↳ |
|   -Selection_9(Build) | 1.25    | cop[tikv] |                   |

```

```

    ↵ not(isnull(test.t1.id))
    ↵
    ↵ |
    |   - IndexRangeScan_7      | 1.25  | cop[tikv] | table:t1, index:id(id
    ↵ ) | range: decided by [eq(test.t1.id, test.t2.id)], keep order:false,
    ↵ stats:pseudo           |
    |   - TableRowIDScan_8(Probe) | 1.25  | cop[tikv] | table:t1
    ↵ keep order:false, stats:pseudo
    ↵
    ↵ |
+---+
    ↵ -----+
    ↵
8 rows in set (0.00 sec)

```

From example 4, you can see that in `dynamic` mode, the execution plan with `IndexJoin` is selected when you execute the query.

11.5.14 Temporary Tables

The temporary tables feature is introduced in TiDB v5.3.0. This feature solves the issue of temporarily storing the intermediate results of an application, which frees you from frequently creating and dropping tables. You can store the intermediate calculation data in temporary tables. When the intermediate data is no longer needed, TiDB automatically cleans up and recycles the temporary tables. This avoids user applications being too complicated, reduces table management overhead, and improves performance.

This document introduces the user scenarios and the types of temporary tables, provides usage examples and instruction on how to limit the memory usage of temporary tables, and explains compatibility restrictions with other TiDB features.

11.5.14.1 User scenarios

You can use TiDB temporary tables in the following scenarios:

- Cache the intermediate temporary data of an application. After the calculation is completed, the data is dumped to the ordinary table, and the temporary table is automatically released.
- Perform multiple DML operations on the same data in a short period of time. For example, in an e-commerce shopping cart application, add, modify, and delete products, complete the payment, and remove the shopping cart information.
- Quickly import intermediate temporary data in batches to improve the performance of importing temporary data.
- Update data in batches. Import data into temporary tables in the database in batches, and export the data to files after finishing modify the data.

11.5.14.2 Types of temporary tables

Temporary tables in TiDB are divided into two types: local temporary tables and global temporary tables.

- For a local temporary table, the table definition and data in the table are visible only to the current session. This type is suitable for temporarily storing intermediate data in the session.
- For a global temporary table, the table definition is visible to the entire TiDB cluster, and the data in the table is visible only to the current transaction. This type is suitable for temporarily storing intermediate data in the transaction.

11.5.14.3 Local temporary tables

The semantics of the local temporary table in TiDB is consistent with that of the MySQL temporary table. The characteristics are as follows:

- The table definition of a local temporary table is not persistent. A local temporary table is visible only to the session in which the table is created, and other sessions cannot access the table.
- You can create local temporary tables with the same name in different sessions, and each session reads only from and writes only to the local temporary table created in the session.
- The data of a local temporary table is visible to all transactions in the session.
- After a session ends, the local temporary table created in the session is automatically dropped.
- A local temporary table can have the same name as an ordinary table. In this case, in the DDL and DML statements, the ordinary table is hidden until the local temporary table is dropped.

To create a local temporary table, you can use the `CREATE TEMPORARY TABLE` statement. To drop a local temporary table, you can use the `DROP TABLE` or `DROP TEMPORARY TABLE` statement.

Different from MySQL, the local temporary tables in TiDB are all external tables, and no internal temporary tables will be created automatically when SQL statements are executed.

11.5.14.3.1 Usage examples of local temporary tables

Note:

- Before you use the temporary table in TiDB, pay attention to the compatibility restrictions with other TiDB features and the compatibility with MySQL temporary tables.

- If you have created local temporary tables on a cluster earlier than TiDB v5.3.0, these tables are actually ordinary tables, and treated as ordinary tables after the cluster is upgraded to TiDB v5.3.0 or a later version.

Assume that there is an ordinary table `users`:

```
CREATE TABLE users (
    id BIGINT,
    name VARCHAR(100),
    PRIMARY KEY(id)
);
```

In session A, creating a local temporary table `users` does not conflict with the ordinary table `users`. When session A accesses the `users` table, it accesses the local temporary table `users`.

```
CREATE TEMPORARY TABLE users (
    id BIGINT,
    name VARCHAR(100),
    city VARCHAR(50),
    PRIMARY KEY(id)
);
```

```
Query OK, 0 rows affected (0.01 sec)
```

If you insert data into `users`, data is inserted to the local temporary table `users` in session A.

```
INSERT INTO users(id, name, city) VALUES(1001, 'Davis', 'LosAngeles');
```

```
Query OK, 1 row affected (0.00 sec)
```

```
SELECT * FROM users;
```

id	name	city
1001	Davis	LosAngeles

```
1 row in set (0.00 sec)
```

In session B, creating a local temporary table `users` does not conflict with the ordinary table `users` or the local temporary table `users` in session A. When session B accesses the `users` table, it accesses the local temporary table `users` in session B.

```
CREATE TEMPORARY TABLE users (
    id BIGINT,
    name VARCHAR(100),
    city VARCHAR(50),
    PRIMARY KEY(id)
);
```

```
Query OK, 0 rows affected (0.01 sec)
```

If you insert data into `users`, data is inserted to the local temporary table `users` in session B.

```
INSERT INTO users(id, name, city) VALUES(1001, 'James', 'NewYork');
```

```
Query OK, 1 row affected (0.00 sec)
```

```
SELECT * FROM users;
```

```
+----+----+----+
| id | name | city |
+----+----+----+
| 1001 | James | NewYork |
+----+----+----+
1 row in set (0.00 sec)
```

11.5.14.3.2 Compatibility with MySQL temporary tables

The following features and limitations of TiDB local temporary tables are the same with those of MySQL temporary tables:

- When you create or drop local temporary tables, the current transaction is not automatically committed.
- After dropping the schema where a local temporary table is located, the temporary table is not dropped and is still readable and writable.
- Creating a local temporary table requires the `CREATE TEMPORARY TABLES` permission. All subsequent operations on the table do not require any permission.
- Local temporary tables do not support foreign keys and partitioned tables.
- Does not support creating views based on local temporary tables.
- `SHOW [FULL] TABLES` does not show local temporary tables.

Local temporary tables in TiDB are incompatible with MySQL temporary tables in the following aspects:

- TiDB local temporary tables do not support `ALTER TABLE`.

- TiDB local temporary tables ignore the `ENGINE` table option, and always store temporary table data in TiDB memory with a [memory limit](#).
- When `MEMORY` is declared as the storage engine, TiDB local temporary tables are not restricted by the `MEMORY` storage engine.
- When `INNODB` or `MYISAM` is declared as the storage engine, TiDB local temporary tables ignore the system variables specific to the InnoDB temporary tables.
- MySQL does not permit referencing to the same temporary table multiple times in the same SQL statement. TiDB local temporary tables do not have this restriction.
- The system table `information_schema.INNODB_TEMP_TABLE_INFO` that shows temporary tables in MySQL does not exist in TiDB. Currently, TiDB does not have a system table that shows local temporary tables.
- TiDB does not have internal temporary tables. The MySQL system variables for internal temporary tables do not take effect for TiDB.

11.5.14.4 Global temporary tables

The global temporary table is an extension of TiDB. The characteristics are as follows:

- The table definition of a global temporary table is persistent and visible to all sessions.
- The data of a global temporary table is visible only in the current transaction. When the transaction ends, the data is automatically cleared.
- A global temporary table cannot have the same name as an ordinary table.

To create a global temporary table, you can use the `CREATE GLOBAL TEMPORARY TABLE` statement ended with `ON COMMIT DELETE ROWS`. To drop a global temporary table, you can use the `DROP TABLE` or `DROP GLOBAL TEMPORARY TABLE` statement.

11.5.14.4.1 Usage examples of global temporary tables

Note:

- Before you use the temporary table in TiDB, pay attention to the [compatibility restrictions with other TiDB features](#).
- If you have created global temporary tables on a TiDB cluster of v5.3.0 or later, when the cluster is downgraded to a version earlier than v5.3.0, these tables are handled as ordinary tables. In this case, a data error occurs.

Create a global temporary table `users` in session A:

```
CREATE GLOBAL TEMPORARY TABLE users (
    id BIGINT,
    name VARCHAR(100),
    city VARCHAR(50),
    PRIMARY KEY(id)
) ON COMMIT DELETE ROWS;
```

Query OK, 0 rows affected (0.01 sec)

The data written to `users` is visible to the current transaction:

```
BEGIN;
```

Query OK, 0 rows affected (0.00 sec)

```
INSERT INTO users(id, name, city) VALUES(1001, 'Davis', 'LosAngeles');
```

Query OK, 1 row affected (0.00 sec)

```
SELECT * FROM users;
```

id	name	city
1001	Davis	LosAngeles

1 row in set (0.00 sec)

After the transaction ends, the data is automatically cleared:

```
COMMIT;
```

Query OK, 0 rows affected (0.00 sec)

```
SELECT * FROM users;
```

Empty set (0.00 sec)

After `users` is created in session A, session B can also read from and write to the `users` table:

```
SELECT * FROM users;
```

Empty set (0.00 sec)

Note:

If the transaction is automatically committed, after the SQL statement is executed, the inserted data is automatically cleared and unavailable to subsequent SQL executions. Therefore, you should use non-autocommit transactions to read from and write to global temporary tables.

11.5.14.5 Limit the memory usage of temporary tables

No matter which storage engine is declared as `ENGINE` when you define a table, the data of local temporary tables and global temporary tables is only stored in the memory of TiDB instances. This data is not persisted.

To avoid memory overflow, you can limit the size of each temporary table using the `tidb_tmp_table_max_size` system variable. Once a temporary table is larger than the `tidb_tmp_table_max_size` threshold value, TiDB reports an error. The default value of `tidb_tmp_table_max_size` is 64MB.

For example, set the maximum size of a temporary table to 256MB:

```
SET GLOBAL tidb_tmp_table_max_size=268435456;
```

11.5.14.6 Compatibility restrictions with other TiDB features

Local temporary tables and global temporary tables in TiDB are **NOT** compatible with the following TiDB features:

- `AUTO_RANDOM` columns
- `SHARD_ROW_ID_BITS` and `PRE_SPLIT_REGIONS` table options
- Partitioned tables
- `SPLIT REGION` statements
- `ADMIN CHECK TABLE` and `ADMIN CHECKSUM TABLE` statements
- `FLASHBACK TABLE` and `RECOVER TABLE` statements
- Executing `CREATE TABLE LIKE` statements based on a temporary table
- Stale Read
- Foreign keys
- SQL bindings
- TiFlash replicas
- Creating views on a temporary table
- Placement Rules
- Execution plans involving a temporary table are not cached by `prepare plan cache`.

Local temporary tables in TiDB do **NOT** support the following feature:

- Reading historical data using the `tidb_snapshot` system variable.

11.5.14.7 TiDB ecosystem tool support

Local temporary tables are not exported, backed up or replicated by TiDB ecosystem tools, because these tables are visible only to the current session.

Global temporary tables are exported, backed up, and replicated by TiDB ecosystem tools, because the table definition is globally visible. Note that the data on the tables are not exported.

Note:

- Replicating temporary tables using TiCDC requires TiCDC v5.3.0 or later. Otherwise, the table definition of the downstream table is wrong.
- Backing up temporary tables using BR requires BR v5.3.0 or later. Otherwise, the table definitions of the backed temporary tables are wrong.
- The cluster to export, the cluster after data restore, and the downstream cluster of a replication should support global temporary tables. Otherwise, an error is reported.

11.5.14.8 See also

- [CREATE TABLE](#)
- [CREATE TABLE LIKE](#)
- [DROP TABLE](#)

11.5.15 Character Set and Collation

This document introduces the character sets and collations supported by TiDB.

11.5.15.1 Concepts

A character set is a set of symbols and encodings. The default character set in TiDB is utf8mb4, which matches the default in MySQL 8.0 and above.

A collation is a set of rules for comparing characters in a character set, and the sorting order of characters. For example in a binary collation A and a do not compare as equal:

```
SET NAMES utf8mb4 COLLATE utf8mb4_bin;
SELECT 'A' = 'a';
SET NAMES utf8mb4 COLLATE utf8mb4_general_ci;
SELECT 'A' = 'a';
```

```

mysql> SELECT 'A' = 'a';
+-----+
| 'A' = 'a' |
+-----+
|      0 |
+-----+
1 row in set (0.00 sec)

mysql> SET NAMES utf8mb4 COLLATE utf8mb4_general_ci;
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT 'A' = 'a';
+-----+
| 'A' = 'a' |
+-----+
|      1 |
+-----+
1 row in set (0.00 sec)

```

TiDB defaults to using a binary collation. This differs from MySQL, which uses a case-insensitive collation by default.

11.5.15.2 Character sets and collations supported by TiDB

Currently, TiDB supports the following character sets:

```
SHOW CHARACTER SET;
```

Charset	Description	Default collation	Maxlen
utf8	UTF-8 Unicode	utf8_bin	3
utf8mb4	UTF-8 Unicode	utf8mb4_bin	4
ascii	US ASCII	ascii_bin	1
latin1	Latin1	latin1_bin	1
binary	binary	binary	1

5 rows in set (0.00 sec)

TiDB supports the following collations:

```
mysql> show collation;
+-----+-----+-----+-----+-----+
| Collation | Charset | Id | Default | Compiled | Sortlen |
+-----+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+-----+
| utf8mb4_bin | utf8mb4 | 46 | Yes | Yes | 1 |
| latin1_bin | latin1 | 47 | Yes | Yes | 1 |
| binary     | binary  | 63 | Yes | Yes | 1 |
| ascii_bin  | ascii   | 65 | Yes | Yes | 1 |
| utf8_bin   | utf8    | 83 | Yes | Yes | 1 |
+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)
```

Warning:

TiDB incorrectly treats latin1 as a subset of utf8. This can lead to unexpected behaviors when you store characters that differ between latin1 and utf8 encodings. It is strongly recommended to the utf8mb4 character set. See [TiDB #18955](#) for more details.

Note:

The default collations in TiDB (binary collations, with the suffix _bin) are different than [the default collations in MySQL](#) (typically general collations, with the suffix _general_ci). This can cause incompatible behavior when specifying an explicit character set but relying on the implicit default collation to be chosen.

You can use the following statement to view the collations (under the [new framework for collations](#)) that corresponds to the character set.

```
SHOW COLLATION WHERE Charset = 'utf8mb4';
```

```
+-----+-----+-----+-----+-----+
| Collation      | Charset | Id | Default | Compiled | Sortlen |
+-----+-----+-----+-----+-----+
| utf8mb4_bin    | utf8mb4 | 46 | Yes    | Yes    | 1 |
| utf8mb4_general_ci | utf8mb4 | 45 |        | Yes    | 1 |
| utf8mb4_unicode_ci | utf8mb4 | 224 |       | Yes    | 1 |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

11.5.15.3 utf8 and utf8mb4 in TiDB

In MySQL, the character set `utf8` is limited to a maximum of three bytes. This is sufficient to store characters in the Basic Multilingual Plane (BMP), but not enough to store characters such as emojis. For this, it is recommended to use the character set `utf8mb4` instead.

By default, TiDB provides the same 3-byte limit on `utf8` to ensure that data created in TiDB can still safely be restored in MySQL. This can be disabled by changing the value of `check-mb4-value-in-utf8` to `FALSE` in your TiDB configuration file.

The following demonstrates the default behavior when inserting a 4-byte emoji character into a table. The `INSERT` statement fails for the `utf8` character set, but succeeds for `utf8mb4`:

```
mysql> CREATE TABLE utf8_test (
    -> c char(1) NOT NULL
    -> ) CHARACTER SET utf8;
Query OK, 0 rows affected (0.09 sec)

mysql> CREATE TABLE utf8m4_test (
    -> c char(1) NOT NULL
    -> ) CHARACTER SET utf8mb4;
Query OK, 0 rows affected (0.09 sec)

mysql> INSERT INTO utf8_test VALUES ('\u2603');
ERROR 1366 (HY000): incorrect utf8 value f09f9889(\u2603) for column c
mysql> INSERT INTO utf8m4_test VALUES ('\u2603');
Query OK, 1 row affected (0.02 sec)

mysql> SELECT char_length(c), length(c), c FROM utf8_test;
Empty set (0.01 sec)

mysql> SELECT char_length(c), length(c), c FROM utf8m4_test;
+-----+-----+-----+
| char_length(c) | length(c) | c   |
+-----+-----+-----+
|           1   |       4   | \u2603 |
+-----+-----+-----+
1 row in set (0.00 sec)
```

11.5.15.4 Character set and collation in different layers

The character set and collation can be set at different layers.

11.5.15.4.1 Database character set and collation

Each database has a character set and a collation. You can use the following statements to specify the database character set and collation:

```
CREATE DATABASE db_name
  [[DEFAULT] CHARACTER SET charset_name]
  [[DEFAULT] COLLATE collation_name]

ALTER DATABASE db_name
  [[DEFAULT] CHARACTER SET charset_name]
  [[DEFAULT] COLLATE collation_name]
```

DATABASE can be replaced with SCHEMA here.

Different databases can use different character sets and collations. Use the `character_set_database` and `collation_database` to see the character set and collation of the current database:

```
CREATE SCHEMA test1 CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
```

```
Query OK, 0 rows affected (0.09 sec)
```

```
USE test1;
```

```
Database changed
```

```
SELECT @@character_set_database, @@collation_database;
```

```
+-----+-----+
| @character_set_database | @@collation_database |
+-----+-----+
| utf8mb4                | utf8mb4_general_ci |
+-----+-----+
1 row in set (0.00 sec)
```

```
CREATE SCHEMA test2 CHARACTER SET latin1 COLLATE latin1_bin;
```

```
Query OK, 0 rows affected (0.09 sec)
```

```
USE test2;
```

```
Database changed
```

```
SELECT @@character_set_database, @@collation_database;
```

```
+-----+-----+
| @@character_set_database | @@collation_database |
+-----+-----+
| latin1                   | latin1_bin           |
+-----+-----+
1 row in set (0.00 sec)
```

You can also see the two values in INFORMATION_SCHEMA:

```
SELECT DEFAULT_CHARACTER_SET_NAME, DEFAULT_COLLATION_NAME
FROM INFORMATION_SCHEMA.SCHEMATA WHERE SCHEMA_NAME = 'db_name';
```

11.5.15.4.2 Table character set and collation

You can use the following statement to specify the character set and collation for tables:

```
CREATE TABLE tbl_name (column_list)
[[DEFAULT] CHARACTER SET charset_name]
[COLLATE collation_name]

ALTER TABLE tbl_name
[[DEFAULT] CHARACTER SET charset_name]
[COLLATE collation_name]
```

For example:

```
CREATE TABLE t1(a int) CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
```

```
Query OK, 0 rows affected (0.08 sec)
```

If the table character set and collation are not specified, the database character set and collation are used as their default values.

11.5.15.4.3 Column character set and collation

You can use the following statement to specify the character set and collation for columns:

```
col_name {CHAR | VARCHAR | TEXT} (col_length)
[CHARACTER SET charset_name]
[COLLATE collation_name]

col_name {ENUM | SET} (val_list)
[CHARACTER SET charset_name]
[COLLATE collation_name]
```

If the column character set and collation are not specified, the table character set and collation are used as their default values.

11.5.15.4.4 String character sets and collation

Each string corresponds to a character set and a collation. When you use a string, this option is available:

`[_charset_name] 'string' [COLLATE collation_name]`

Example:

```
SELECT 'string';
SELECT _utf8mb4 'string';
SELECT _utf8mb4 'string' COLLATE utf8mb4_general_ci;
```

Rules:

- Rule 1: If you specify `CHARACTER SET charset_name` and `COLLATE collation_name`, then the `charset_name` character set and the `collation_name` collation are used directly.
- Rule 2: If you specify `CHARACTER SET charset_name` but do not specify `COLLATE collation_name`, the `charset_name` character set and the default collation of `charset_name` are used.
- Rule 3: If you specify neither `CHARACTER SET charset_name` nor `COLLATE collation_name`, the character set and collation given by the system variables `character_set_connection` and `collation_connection` are used.

11.5.15.4.5 Client connection character set and collation

- The server character set and collation are the values of the `character_set_server` and `collation_server` system variables.
- The character set and collation of the default database are the values of the `character_set_database` and `collation_database` system variables.

You can use `character_set_connection` and `collation_connection` to specify the character set and collation for each connection. The `character_set_client` variable is to set the client character set.

Before returning the result, the `character_set_results` system variable indicates the character set in which the server returns query results to the client, including the metadata of the result.

You can use the following statement to set the character set and collation that is related to the client:

- `SET NAMES 'charset_name' [COLLATE 'collation_name']`

`SET NAMES` indicates what character set the client will use to send SQL statements to the server. `SET NAMES utf8mb4` indicates that all the requests from the client use `utf8mb4`, as well as the results from the server.

The `SET NAMES 'charset_name'` statement is equivalent to the following statement combination:

```
SET character_set_client = charset_name;
SET character_set_results = charset_name;
SET character_set_connection = charset_name;
```

`COLLATE` is optional, if absent, the default collation of the `charset_name` is used to set the `collation_connection`.

- `SET CHARACTER SET 'charset_name'`

Similar to `SET NAMES`, the `SET NAMES 'charset_name'` statement is equivalent to the following statement combination:

```
SET character_set_client = charset_name;
SET character_set_results = charset_name;
SET charset_connection = @@charset_database;
SET collation_connection = @@collation_database;
```

11.5.15.5 Selection priorities of character sets and collations

String > Column > Table > Database > Server

11.5.15.6 General rules on selecting character sets and collation

- Rule 1: If you specify `CHARACTER SET charset_name` and `COLLATE collation_name` , then the `charset_name` character set and the `collation_name` collation are used directly.
- Rule 2: If you specify `CHARACTER SET charset_name` and do not specify `COLLATE collation_name`, then the `charset_name` character set and the default collation of `charset_name` are used.
- Rule 3: If you specify neither `CHARACTER SET charset_name` nor `COLLATE collation_name`, the character set and collation with higher optimization levels are used.

11.5.15.7 Validity check of characters

If the specified character set is `utf8` or `utf8mb4`, TiDB only supports the valid `utf8` characters. For invalid characters, TiDB reports the `incorrect utf8 value` error. This validity check of characters in TiDB is compatible with MySQL 8.0 but incompatible with MySQL 5.7 or earlier versions.

To disable this error reporting, use `set @@tidb_skip_utf8_check=1;` to skip the character check.

11.5.15.8 Collation support framework

The syntax support and semantic support for the collation are influenced by the `new_collations_enabled_on_first_bootstrap` configuration item. The syntax support and semantic support are different. The former indicates that TiDB can parse and set collations. The latter indicates that TiDB can correctly use collations when comparing strings.

Before v4.0, TiDB provides only the [old framework for collations](#). In this framework, TiDB supports syntactically parsing most of the MySQL collations but semantically takes all collations as binary collations.

Since v4.0, TiDB supports a [new framework for collations](#). In this framework, TiDB semantically parses different collations and strictly follows the collations when comparing strings.

11.5.15.8.1 Old framework for collations

Before v4.0, you can specify most of the MySQL collations in TiDB, and these collations are processed according to the default collations, which means that the byte order determines the character order. Different from MySQL, TiDB deletes the space at the end of the character according to the PADDING attribute of the collation before comparing characters, which causes the following behavior differences:

```
CREATE TABLE t(a varchar(20) charset utf8mb4 collate utf8mb4_general_ci
    ↪ PRIMARY KEY);
Query OK, 0 rows affected
INSERT INTO t VALUES ('A');
Query OK, 1 row affected
INSERT INTO t VALUES ('a');
Query OK, 1 row affected # In TiDB, it is successfully executed. In MySQL,
    ↪ because utf8mb4_general_ci is case-insensitive, the `Duplicate entry
    ↪ 'a'` error is reported.
INSERT INTO t1 VALUES ('a ');
Query OK, 1 row affected # In TiDB, it is successfully executed. In MySQL,
    ↪ because comparison is performed after the spaces are filled in, the
    ↪ Duplicate entry 'a '` error is returned.
```

11.5.15.8.2 New framework for collations

In TiDB 4.0, a complete framework for collations is introduced. This new framework supports semantically parsing collations and introduces the `new_collations_enabled_on_first_bootstrap` configuration item to decide whether to enable the new framework when a cluster is first initialized. If you initialize the cluster after the configuration item is enabled, you can check whether the new collation is enabled through the `new_collation_enabled` variable in the `mysql.tidb` table:

```
SELECT VARIABLE_VALUE FROM mysql.tidb WHERE VARIABLE_NAME='
↪ new_collation_enabled';
```

VARIABLE_VALUE
True

1 row in set (0.00 sec)

Under the new framework, TiDB support the utf8_general_ci, utf8mb4_general_ci, utf8_unicode_ci, and utf8mb4_unicode_ci collations which are compatible with MySQL.

When one of utf8_general_ci, utf8mb4_general_ci, utf8_unicode_ci, and utf8mb4_unicode_ci is used, the string comparison is case-insensitive and accent-insensitive. At the same time, TiDB also corrects the collation's PADDING behavior:

```
CREATE TABLE t(a varchar(20) charset utf8mb4 collate utf8mb4_general_ci
↪ PRIMARY KEY);
Query OK, 0 rows affected (0.00 sec)
INSERT INTO t VALUES ('A');
Query OK, 1 row affected (0.00 sec)
INSERT INTO t VALUES ('a');
ERROR 1062 (23000): Duplicate entry 'a' for key 'PRIMARY' # TiDB is
↪ compatible with the case-insensitive collation of MySQL.
INSERT INTO t VALUES ('a ');
ERROR 1062 (23000): Duplicate entry 'a ' for key 'PRIMARY' # TiDB modifies
↪ the `PADDING` behavior to be compatible with MySQL.
```

Note:

The implementation of padding in TiDB is different from that in MySQL. In MySQL, padding is implemented by filling in spaces. In TiDB, padding is implemented by cutting out the spaces at the end. The two approaches are the same in most cases. The only exception is when the end of the string contains characters that are less than spaces (0x20). For example, the result of 'a' < 'a\t' in TiDB is 1, but in MySQL, 'a' < 'a\t' is equivalent to 'a ' < 'a\t', and the result is 0.

11.5.15.9 Coercibility values of collations in expressions

If an expression involves multiple clauses of different collations, you need to infer the collation used in the calculation. The rules are as follows:

- The coercibility value of the explicit `COLLATE` clause is 0.
- If the collations of two strings are incompatible, the coercibility value of the concatenation of two strings with different collations is 1.
- The collation of the column, `CAST()`, `CONVERT()`, or `BINARY()` has a coercibility value of 2.
- The system constant (the string returned by `USER()` or `VERSION()`) has a coercibility value of 3.
- The coercibility value of constants is 4.
- The coercibility value of numbers or intermediate variables is 5.
- `NULL` or expressions derived from `NULL` has a coercibility value of 6.

When inferring collations, TiDB prefers using the collation of expressions with lower coercibility values. If the coercibility values of two clauses are the same, the collation is determined according to the following priority:

`binary > utf8mb4_bin > (utf8mb4_general_ci = utf8mb4_unicode_ci) > utf8_bin > (utf8_general_ci = utf8_unicode_ci) > latin1_bin > ascii_bin`

TiDB cannot infer the collation and reports an error in the following situations:

- If the collations of two clauses are different and the coercibility value of both clauses is 0.
- If the collations of two clauses are incompatible and the returned type of expression is `String`.

11.5.15.10 `COLLATE` clause

TiDB supports using the `COLLATE` clause to specify the collation of an expression. The coercibility value of this expression is 0, which has the highest priority. See the following example:

```
SELECT 'a' = _utf8mb4 'A' collate utf8mb4_general_ci;
```

+-----+	
	'a' = _utf8mb4 'A' collate utf8mb4_general_ci
+-----+	
	1
+-----+	
1 row in set (0.00 sec)	

For more details, see [Connection Character Sets and Collations](#).

11.5.16 Placement Rules in SQL

Warning:

Placement Rules in SQL is an experimental feature introduced in v5.3.0. The syntax might change before its GA, and there might also be bugs. If you understand the risks, you can enable this experiment feature by executing
`SET GLOBAL tidb_enable_alter_placement = 1;.`

Placement Rules in SQL is a feature that enables you to specify where data is stored in a TiKV cluster using SQL interfaces. Using this feature, tables and partitions are scheduled to specific regions, data centers, racks, or hosts. This is useful for scenarios including optimizing a high availability strategy with lower cost, ensuring that local replicas of data are available for local stale reads, and adhering to data locality requirements.

The detailed user scenarios are as follows:

- Merge multiple databases of different applications to reduce the cost on database maintenance
- Increase replica count for important data to improve the application availability and data reliability
- Store new data into SSDs and store old data into HDDs to lower the cost on data archiving and storage
- Schedule the leaders of hotspot data to high-performance TiKV instances
- Separate cold data to lower-cost storage mediums to improve cost efficiency

11.5.16.1 Specify placement options

To use Placement Rules in SQL, you need to specify one or more placement options in a SQL statement. To specify the Placement options, you can either use *direct placement* or use a *placement policy*.

In the following example, both tables `t1` and `t2` have the same rules. `t1` is specified rules using a direct placement while `t2` is specified rules using a placement policy.

```
CREATE TABLE t1 (a INT) PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us-
→ west-1";
CREATE PLACEMENT POLICY eastandwest PRIMARY_REGION="us-east-1" REGIONS="us-
→ east-1,us-west-1";
CREATE TABLE t2 (a INT) PLACEMENT POLICY=eastandwest;
```

It is recommended to use placement policies for simpler rule management. When you change a placement policy (via `ALTER PLACEMENT POLICY`), the change automatically propagates to all database objects.

If you use direct placement options, you have to alter rules for each object (for example, tables and partitions).

`PLACEMENT POLICY` is not associated with any database schema and has the global scope. Therefore, assigning a placement policy does not require any additional privileges over the `CREATE TABLE` privilege.

11.5.16.2 Option reference

Note:

Placement options depend on labels correctly specified in the configuration of each TiKV node. For example, the `PRIMARY_REGION` option depends on the `region` label in TiKV. To see a summary of all labels available in your TiKV cluster, use the statement `SHOW PLACEMENT LABELS`:

```
mysql> show placement labels;
+-----+-----+
| Key   | Values        |
+-----+-----+
| disk  | ["ssd"]       |
| region | ["us-east-1"] |
| zone  | ["us-east-1a"] |
+-----+-----+
3 rows in set (0.00 sec)
```

Option Name	Description
<code>PRIMARY_REGION</code>	Raft leaders are placed in stores that have the <code>region</code> label that matches the value of this option.
<code>REGIONS</code>	Raft followers are placed in stores that have the <code>region</code> label that matches the value of this option.
<code>SCHEDULE</code>	The strategy used to schedule the placement of followers. The value options are <code>EVEN</code> (default) or <code>MAJORITY_IN_PRIMARY</code> .
<code>FOLLOWERS</code>	The number of followers. For example, <code>FOLLOWERS=2</code> means that there will be 3 replicas of the data (2 followers and 1 leader).

In addition to the placement options above, you can also use the advance configurations. For details, see [Advance placement](#).

Option Name	Description
CONSTRAINTS	A list of constraints that apply to all roles. For example, CONSTRAINTS → ="+disk → =ssd].
FOLLOWER_CONSTRAINTS	of constraints that only apply to followers.

11.5.16.3 Examples

11.5.16.3.1 Increase the number of replicas

The default configuration of `max-relicas` is 3. To increase this for a specific set of tables, you can use a placement policy as follows:

```
CREATE PLACEMENT POLICY fivereplicas FOLLOWERS=4;
CREATE TABLE t1 (a INT) PLACEMENT POLICY=fivereplicas;
```

Note that the PD configuration includes the leader and follower count, thus 4 followers + 1 leader equals 5 replicas in total.

To expand on this example, you can also use `PRIMARY_REGION` and `REGIONS` placement options to describe the placement for the followers:

```
CREATE PLACEMENT POLICY eastandwest PRIMARY_REGION="us-east-1" REGIONS="us-
→ east-1,us-east-2,us-west-1" SCHEDULE="MAJORITY_IN_PRIMARY" FOLLOWERS
→ =4;
CREATE TABLE t1 (a INT) PLACEMENT POLICY=eastandwest;
```

The `SCHEDULE` option instructs TiDB on how to balance the followers. The default schedule of `EVEN` ensures a balance of followers in all regions.

To ensure that enough followers are placed in the primary region (`us-east-1`) so that quorum can be achieved, you can use the `MAJORITY_IN_PRIMARY` schedule. This schedule helps provide lower latency transactions at the expense of some availability. If the primary region fails, `MAJORITY_IN_PRIMARY` cannot provide automatic failover.

11.5.16.3.2 Assign placement to a partitioned table

Note:

The following example uses list partitioning, which is currently an experimental feature of TiDB. Partitioned tables also require the PRIMARY KEY to be included in all columns in the table's partitioning function.

In addition to assigning placement options to tables, you can also assign the options to table partitions. For example:

```
CREATE PLACEMENT POLICY europe PRIMARY_REGION="eu-central-1" REGIONS="eu-
    ↪ central-1,eu-west-1";
CREATE PLACEMENT POLICY northamerica PRIMARY_REGION="us-east-1" REGIONS="us
    ↪ -east-1";

SET tidb_enable_list_partition = 1;
CREATE TABLE t1 (
    country VARCHAR(10) NOT NULL,
    userdata VARCHAR(100) NOT NULL
) PARTITION BY LIST COLUMNS (country) (
    PARTITION pEurope VALUES IN ('DE', 'FR', 'GB') PLACEMENT POLICY=europe,
    PARTITION pNorthAmerica VALUES IN ('US', 'CA', 'MX') PLACEMENT POLICY=
        ↪ northamerica
);
```

11.5.16.3.3 Set the default placement for a schema

You can directly attach the default placement options to a database schema. This works similar to setting the default character set or collation for a schema. Your specified placement options apply when no other options are specified. For example:

```
CREATE TABLE t1 (a INT); -- Creates a table t1 with no placement options.

ALTER DATABASE test FOLLOWERS=4; -- Changes the default placement option,
    ↪ and does not apply to the existing table t1.

CREATE TABLE t2 (a INT); -- Creates a table t2 with the default placement
    ↪ of FOLLOWERS=4.

CREATE TABLE t3 (a INT) PRIMARY_REGION="us-east-1" REGIONS="us-east-1,us-
    ↪ east-2"; -- Creates a table t3 without the default FOLLOWERS=4
    ↪ placement, because this statement has specified another placement.
```

```
ALTER DATABASE test FOLLOWERS=2; -- Changes the default placement, and does
→ not apply to existing tables.
```

```
CREATE TABLE t4 (a INT); -- Creates a table t4 with the default FOLLOWERS
→ =2 option.
```

Because placement options are only inherited from the database schema when a table is created, it is recommended to set the default placement option using a **PLACEMENT POLICY**. This ensures that future changes to the policy propagate to existing tables.

11.5.16.3.4 Advanced placement

The placement options **PRIMARY_REGION**, **REGIONS**, and **SCHEDULE** meet the basic needs of data placement at the loss of some flexibility. For more complex scenarios with the need for higher flexibility, you can also use the advanced placement options of **CONSTRAINTS** and **FOLLOWER_CONSTRAINTS**. You cannot specify the **PRIMARY_REGION**, **REGIONS**, or **SCHEDULE** option with the **CONSTRAINTS** option at the same time. If you specify both at the same time, an error will be returned.

For example, to set constraints that data must reside on a TiKV store where the label **disk** must match a value:

```
CREATE PLACEMENT POLICY storeonfastssd CONSTRAINTS="[+disk=ssd]";  

CREATE PLACEMENT POLICY storeonhdd CONSTRAINTS="[+disk=hdd]";  

CREATE PLACEMENT POLICY companystandardpolicy CONSTRAINTS="";  
  

CREATE TABLE t1 (id INT, name VARCHAR(50), purchased DATE)  

PLACEMENT POLICY=companystandardpolicy  

PARTITION BY RANGE( YEAR(purchased) ) (  

    PARTITION p0 VALUES LESS THAN (2000) PLACEMENT POLICY=storeonhdd,  

    PARTITION p1 VALUES LESS THAN (2005),  

    PARTITION p2 VALUES LESS THAN (2010),  

    PARTITION p3 VALUES LESS THAN (2015),  

    PARTITION p4 VALUES LESS THAN MAXVALUE PLACEMENT POLICY=storeonfastssd  

);
```

You can either specify constraints in list format (**[+disk=ssd]**) or in dictionary format (**{+disk=ssd: 1,+disk=hdd: 2}**).

In list format, constraints are specified as a list of key-value pairs. The key starts with either a **+** or a **-**. **+disk=ssd** indicates that the label **disk** must be set to **ssd**, and **-disk=hdd** indicates that the label **disk** must not be **hdd**.

In dictionary format, constraints also indicate a number of instances that apply to that rule. For example, **FOLLOWER_CONSTRAINTS="[{+region=us-east-1: 1,+region=us-east**
→ **-2: 1,+region=us-west-1: 1,+any: 1}]"**; indicates that 1 follower is in us-east-1, 1 follower is in us-east-2, 1 follower is in us-west-1, and 1 follower can be in any region.

For another example, `FOLLOWER_CONSTRAINTS='{"+region=us-east-1,+disk=hdd":1,"+region=us-west-1":1}'`; indicates that 1 follower is in us-east-1 with an hdd disk, and 1 follower is in us-west-1.

Note:

Dictionary and list formats are based on the YAML parser, but the YAML syntax might be incorrectly parsed. For example, "`{+disk=ssd:1,+disk=hdd:2}`" is incorrectly parsed as '`{"+disk=ssd:1": null, "+disk=hdd:2": null}`'. But "`{+disk=ssd: 1,+disk=hdd: 1}`" is correctly parsed as '`{"+disk=ssd": 1, "+disk=hdd": 1}`'.

11.5.16.4 Known limitations

The following known limitations exist in the experimental release of Placement Rules in SQL:

- Dumpling does not support dumping placement policies. See [issue #29371](#).
- TiDB tools, including Backup & Restore (BR), TiCDC, TiDB Lightning, and TiDB Data Migration (DM), do not yet support placement rules.
- Temporary tables do not support placement options (either via direct placement or placement policies).
- Syntactic sugar rules are permitted for setting `PRIMARY_REGION` and `REGIONS`. In the future, we plan to add varieties for `PRIMARY_RACK`, `PRIMARY_ZONE`, and `PRIMARY_HOST`. See [issue #18030](#).
- TiFlash learners are not configurable through Placement Rules syntax.
- Placement rules only ensure that data at rest resides on the correct TiKV store. The rules do not guarantee that data in transit (via either user queries or internal operations) only occurs in a specific region.

11.5.17 System Tables

11.5.17.1 mysql Schema

The `mysql` schema contains TiDB system tables. The design is similar to the `mysql` schema in MySQL, where tables such as `mysql.user` can be edited directly. It also contains a number of tables which are extensions to MySQL.

11.5.17.1.1 Grant system tables

These system tables contain grant information about user accounts and their privileges:

- `user`: user accounts, global privileges, and other non-privilege columns

- db: database-level privileges
- tables_priv: table-level privileges
- columns_priv: column-level privileges

11.5.17.1.2 Server-side help system tables

Currently, the `help_topic` is NULL.

11.5.17.1.3 Statistics system tables

- stats_buckets: the buckets of statistics
- stats_histograms: the histograms of statistics
- stats_meta: the meta information of tables, such as the total number of rows and updated rows

11.5.17.1.4 GC worker system tables

- gc_delete_range: to record the data to be deleted

11.5.17.1.5 Miscellaneous system tables

- GLOBAL_VARIABLES: global system variable table
- tidb: to record the version information when TiDB executes `bootstrap`

11.5.17.2 INFORMATION_SCHEMA

11.5.17.2.1 Information Schema

Information Schema provides an ANSI-standard way of viewing system metadata. TiDB also provides a number of custom INFORMATION_SCHEMA tables, in addition to the tables included for MySQL compatibility.

Many INFORMATION_SCHEMA tables have a corresponding SHOW command. The benefit of querying INFORMATION_SCHEMA is that it is possible to join between tables.

Tables for MySQL compatibility

Table Name	Description
<code>CHARACTER_SETS</code>	Provides a list of character sets the server supports.
<code>COLLATIONS</code>	Provides a list of collations that the server supports.

Table Name	Description
COLLATION_CHARACTER_SET_APPLICABILITY	Explains which collations apply to which character sets.
COLUMNS	Provides a list of columns for all tables.
COLUMN_PRIVILEGES	Not implemented by TiDB. Returns zero rows.
COLUMN_STATISTICS	Not implemented by TiDB. Returns zero rows.
ENGINES	Provides a list of supported storage engines.
EVENTS	Not implemented by TiDB. Returns zero rows.
FILES	Not implemented by TiDB. Returns zero rows.
GLOBAL_STATUS	Not implemented by TiDB. Returns zero rows.
GLOBAL_VARIABLES	Not implemented by TiDB. Returns zero rows.
KEY_COLUMN_USAGE	Describes the key constraints of the columns, such as the primary key constraint.
OPTIMIZER_TRACE	Not implemented by TiDB. Returns zero rows.
PARAMETERS	Not implemented by TiDB. Returns zero rows.
PARTITIONS	Provides a list of table partitions.
PLUGINS	Not implemented by TiDB. Returns zero rows.

Table Name	Description
PROCESSLIST	Provides similar information to the command <code>SHOW PROCESSLIST</code> . Not implemented by TiDB. Returns zero rows.
PROFILING	Not implemented by TiDB. Returns zero rows.
REFERENTIAL_CONSTRAINTS	Provides information on <code>FOREIGN KEY</code> constraints.
ROUTINES	Not implemented by TiDB. Returns zero rows.
SCHEMATA	Provides similar information to <code>SHOW DATABASES</code> .
SCHEMA_PRIVILEGES	Not implemented by TiDB. Returns zero rows.
SESSION_STATUS	Not implemented by TiDB. Returns zero rows.
SESSION_VARIABLES	Provides similar functionality to the command <code>SHOW SESSION → VARIABLES</code>
STATISTICS	Provides information on table indexes.
TABLES	Provides a list of tables that the current user has visibility of. Similar to <code>SHOW TABLES</code> .
TABLESPACES	Not implemented by TiDB. Returns zero rows.
TABLE_CONSTRAINTS	Provides information on primary keys, unique indexes and foreign keys.

Table Name	Description
TABLE_PRIVILEGES	Not implemented by TiDB. Returns zero rows.
TRIGGERS	Not implemented by TiDB. Returns zero rows.
USER_PRIVILEGES	Summarizes the privileges associated with the current user.
VIEWS	Provides a list of views that the current user has visibility of. Similar to running SHOW FULL TABLES → WHERE → table_type = ' → VIEW'

Tables that are TiDB extensions

Table Name	Description
ANALYZE_STATUS	Provides information about tasks to collect statistics.
CLIENT_ERRORS_SUMMARY_BY_HOST	Provides a summary of errors and warnings generated by client requests and returned to clients.
CLIENT_ERRORS_SUMMARY_BY_USER	Provides a summary of errors and warnings generated by clients.

Table Name	Description
<code>CLIENT_ERRORS_SUMMARY_GLOBAL</code>	Provides a summary of errors and warnings generated by clients.
<code>CLUSTER_CONFIG</code>	Provides details about configuration settings for the entire TiDB cluster.
<code>CLUSTER_DEADLOCKS</code>	Provides a cluster-level view of the DEADLOCKS table.
<code>CLUSTER_HARDWARE</code>	Provides details on the underlying physical hardware discovered on each TiDB component.
<code>CLUSTER_INFO</code>	Provides details on the current cluster topology.
<code>CLUSTER_LOAD</code>	Provides current load information for TiDB servers in the cluster.

Table Name	Description
CLUSTER_LOG	Provides a log for the entire TiDB cluster
CLUSTER_PROCESSLIST	Provides a cluster-level view of the PROCESSLIST ↪ table.
CLUSTER_SLOW_QUERY	Provides a cluster-level view of the SLOW_QUERY ↪ table.
CLUSTER_STATEMENTS_SUMMARY	Provides a cluster-level view of the STATEMENTS_SUMMARY ↪ table.
CLUSTER_STATEMENTS_SUMMARY_HISTORY	Provides a cluster-level view of the STATEMENTS_SUMMARY_HISTORY ↪ table.
CLUSTER_TIDB_TRX	Provides a cluster-level view of the TIDB_TRX table.
CLUSTER_SYSTEMINFO	Provides details about kernel parameter configuration for servers in the cluster.

Table Name	Description
<code>DATA_LOCK_WAITS</code>	Provides the lock-waiting information on the TiKV server.
<code>DDL_JOBS</code>	Provides similar output to <code>ADMIN ↗ SHOW</code> <code>ADMIN ↗ DDL</code> <code>ADMIN ↗ JOBS</code>
<code>DEADLOCKS</code>	Provides the information of several deadlock errors that have recently occurred.
<code>INSPECTION_RESULT</code>	Triggers internal diagnostics checks.
<code>INSPECTION_RULES</code>	A list of internal diagnostic checks performed.
<code>INSPECTION_SUMMARY</code>	A summarized report of important monitoring metrics.
<code>METRICS_SUMMARY</code>	A summary of metrics extracted from Prometheus.

Table Name	Description
METRICS_SUMMARY_BY_LABEL	See METRICS_SUMMARY ↪ table.
METRICS_TABLES	Provides the PromQL definitions for tables in METRICS_SCHEMA ↪ .
PLACEMENT_RULES	Provides information on all objects that have explicit placement rules assigned.
SEQUENCES	The TiDB implementation of sequences is based on MariaDB.
SLOW_QUERY	Provides information on slow queries on the current TiDB server.
STATEMENTS_SUMMARY	Similar to performance_schema statement summary in MySQL.

Table Name	Description
<code>STATEMENTS_SUMMARY_HISTORY</code>	Similar to performance_schema statement summary history in MySQL.
<code>TABLE_STORAGE_STATS</code>	Provides details about table sizes in storage.
<code>TIDB_HOT_REGIONS</code>	Provides statistics about which regions are hot.
<code>TIDB_INDEXES</code>	Provides index information about TiDB tables.
<code>TIDB_SERVERS_INFO</code>	Provides a list of TiDB servers (namely, tidb-server component)
<code>TIDB_TRX</code>	Provides the information of the transactions that are being executed on the TiDB node.

Table Name	Description
TIFLASH_REPLICA	Provides details about TiFlash replicas.
TIKV_REGION_PEERS	Provides details about where regions are stored.
TIKV_REGION_STATUS	Provides statistics about regions.
TIKV_STORE_STATUS	Provides basic information about TiKV servers.

11.5.17.2.2 ANALYZE_STATUS

The ANALYZE_STATUS table provides information about the running tasks that collect statistics and a limited number of history tasks.

```
USE information_schema;
DESC analyze_status;
```

Field	Type	Null	Key	Default	Extra
TABLE_SCHEMA	varchar(64)	YES		NULL	
TABLE_NAME	varchar(64)	YES		NULL	
PARTITION_NAME	varchar(64)	YES		NULL	
JOB_INFO	varchar(64)	YES		NULL	
PROCESSED_ROWS	bigint(20) unsigned	YES		NULL	
START_TIME	datetime	YES		NULL	
STATE	varchar(64)	YES		NULL	

7 rows in set (0.00 sec)

```
SELECT * FROM `ANALYZE_STATUS`;
```

TABLE_SCHEMA	TABLE_NAME	PARTITION_NAME	JOB_INFO	PROCESSED_ROWS
START_TIME	STATE			
test	t		analyze index idx	2
↪ 2019-06-21 19:51:14	↪ finished			
test	t		analyze columns	2
↪ 2019-06-21 19:51:14	↪ finished			
test	t1	p0	analyze columns	0
↪ 2019-06-21 19:51:15	↪ finished			
test	t1	p3	analyze columns	0
↪ 2019-06-21 19:51:15	↪ finished			
test	t1	p1	analyze columns	0
↪ 2019-06-21 19:51:15	↪ finished			
test	t1	p2	analyze columns	1
↪ 2019-06-21 19:51:15	↪ finished			

6 rows in set

Fields in the ANALYZE_STATUS table are described as follows:

- TABLE_SCHEMA: The name of the database to which the table belongs.
- TABLE_NAME: The name of the table.
- PARTITION_NAME: The name of the partitioned table.
- JOB_INFO: The information of the ANALYZE task.
- PROCESSED_ROWS: The number of rows that have been processed.
- START_TIME: The start time of the ANALYZE task.
- STATE: The execution status of the ANALYZE task. Its value can be pending, running, ↪ ,finished or failed.

11.5.17.2.3 CLIENT_ERRORS_SUMMARY_BY_HOST

The table CLIENT_ERRORS_SUMMARY_BY_HOST provides a summary of SQL errors and warnings that have been returned to clients that connect to a TiDB server. These include:

- Malformed SQL statements.
- Division by zero errors.
- The attempt to insert out-of-range or duplicate key values.
- Permission errors.

- A table that does not exist.

These errors are returned to the client via the MySQL server protocol, where applications are expected to take appropriate action. The `information_schema` \hookrightarrow `.CLIENT_ERRORS_SUMMARY_BY_HOST` table provides a useful method to inspect errors in the scenario where applications are not correctly handling (or logging) errors returned by the TiDB server.

Because `CLIENT_ERRORS_SUMMARY_BY_HOST` summarizes the errors on a per-remote-host basis, it can be useful to diagnose scenarios where one application server is generating more errors than other servers. Possible scenarios include:

- An outdated MySQL client library.
- An outdated application (possibly this server was missed when rolling out a new deployment).
- Incorrect usage of the “host” portion of user permissions.
- Unreliable network connectivity generating more timeouts or disconnected connections.

The summarized counts can be reset using the statement `FLUSH CLIENT_ERRORS_SUMMARY` \hookrightarrow . The summary is local to each TiDB server and is only retained in memory. Summaries will be lost if the TiDB server restarts.

```
USE information_schema;
DESC CLIENT_ERRORS_SUMMARY_BY_HOST;
```

Field	Type	Null	Key	Default	Extra
HOST	varchar(255)	NO		NULL	
ERROR_NUMBER	bigint(64)	NO		NULL	
ERROR_MESSAGE	varchar(1024)	NO		NULL	
ERROR_COUNT	bigint(64)	NO		NULL	
WARNING_COUNT	bigint(64)	NO		NULL	
FIRST_SEEN	timestamp	YES		NULL	
LAST_SEEN	timestamp	YES		NULL	

7 rows in set (0.00 sec)

Field description:

- **HOST**: The remote host of the client.
- **ERROR_NUMBER**: The MySQL-compatible error number that was returned.
- **ERROR_MESSAGE**: The error message which matches the error number (in prepared statement form).
- **ERROR_COUNT**: The number of times this error was returned to the client host.

- `WARNING_COUNT`: The number of times this warning was returned to the client host.
- `FIRST_SEEN`: The first time this error (or warning) was seen from the client host.
- `LAST_SEEN`: The most recent time this error (or warning) was seen from the client host.

The following example shows a warning being generated when the client connects to a local TiDB server. The summary is reset after executing `FLUSH CLIENT_ERRORS_SUMMARY`:

```
SELECT 0/0;
SELECT * FROM CLIENT_ERRORS_SUMMARY_BY_HOST;
FLUSH CLIENT_ERRORS_SUMMARY;
SELECT * FROM CLIENT_ERRORS_SUMMARY_BY_HOST;
```

```
+----+
| 0/0 |
+----+
| NULL |
+----+
1 row in set, 1 warning (0.00 sec)

+-
→ -----+-----+-----+-----+
→
| HOST      | ERROR_NUMBER | ERROR_MESSAGE | ERROR_COUNT | WARNING_COUNT |
→ FIRST_SEEN | LAST_SEEN     |             |
+-
→ -----+-----+-----+-----+
→
| 127.0.0.1 |      1365 | Division by 0 |          0 |          1 |
→ 2021-03-18 12:51:54 | 2021-03-18 12:51:54 |
+-
→ -----+-----+-----+-----+
→
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Empty set (0.00 sec)
```

11.5.17.2.4 CLIENT_ERRORS_SUMMARY_BY_USER

The table `CLIENT_ERRORS_SUMMARY_BY_USER` provides a summary of SQL errors and warnings that have been returned to clients that connect to a TiDB server. These include:

- Malformed SQL statements.

- Division by zero errors.
- The attempt to insert out-of-range or duplicate key values.
- Permission errors.
- A table that does not exist.

Client errors are returned to the client via the MySQL server protocol, where applications are expected to take appropriate action. The `information_schema` → `.CLIENT_ERRORS_SUMMARY_BY_USER` table provides an useful method to inspect errors in the scenario where applications are not correctly handling (or logging) errors returned by the TiDB server.

Because `CLIENT_ERRORS_SUMMARY_BY_USER` summarizes the errors on a per-user basis, it can be useful to diagnose scenarios where one user server is generating more errors than other servers. Possible scenarios include:

- Permission errors.
- Missing tables, or relational objects.
- Incorrect SQL syntax, or incompatibilities between the application and the version of TiDB.

The summarized counts can be reset with the statement `FLUSH CLIENT_ERRORS_SUMMARY` → `.`. The summary is local to each TiDB server and is only retained in memory. Summaries will be lost if the TiDB server restarts.

```
USE information_schema;
DESC CLIENT_ERRORS_SUMMARY_BY_USER;
```

Field	Type	Null	Key	Default	Extra
USER	varchar(64)	NO		NULL	
ERROR_NUMBER	bigint(64)	NO		NULL	
ERROR_MESSAGE	varchar(1024)	NO		NULL	
ERROR_COUNT	bigint(64)	NO		NULL	
WARNING_COUNT	bigint(64)	NO		NULL	
FIRST_SEEN	timestamp	YES		NULL	
LAST_SEEN	timestamp	YES		NULL	

7 rows in set (0.00 sec)

Field description:

- `USER`: The authenticated user.
- `ERROR_NUMBER`: The MySQL-compatible error number that was returned.

- **ERROR_MESSAGE**: The error message which matches the error number (in prepared statement form).
- **ERROR_COUNT**: The number of times this error was returned to the user.
- **WARNING_COUNT**: The number of times this warning was returned to the user.
- **FIRST_SEEN**: The first time this error (or warning) was sent to the user.
- **LAST_SEEN**: The most recent time this error (or warning) was sent to the user.

The following example shows a warning being generated when the client connects to a local TiDB server. The summary is reset after executing `FLUSH CLIENT_ERRORS_SUMMARY`:

```
SELECT 0/0;
SELECT * FROM CLIENT_ERRORS_SUMMARY_BY_USER;
FLUSH CLIENT_ERRORS_SUMMARY;
SELECT * FROM CLIENT_ERRORS_SUMMARY_BY_USER;

+----+
| 0/0 |
+----+
| NULL |
+----+
1 row in set, 1 warning (0.00 sec)

+--+
→ -----+-----+-----+-----+-----+
→
| USER | ERROR_NUMBER | ERROR_MESSAGE | ERROR_COUNT | WARNING_COUNT |
→ FIRST_SEEN | LAST_SEEN      |
+--+
→ -----+-----+-----+-----+-----+
→
| root | 1365 | Division by 0 | 0 | 1 | 2021-03-18
→ 13:05:36 | 2021-03-18 13:05:36 |
+--+
→ -----+-----+-----+-----+-----+
→
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Empty set (0.00 sec)
```

11.5.17.2.5 CLIENT_ERRORS_SUMMARY_GLOBAL

The table `CLIENT_ERRORS_SUMMARY_GLOBAL` provides a global summary of all SQL errors and warnings that have been returned to clients that connect to a TiDB server. These include:

- Malformed SQL statements.
- Division by zero errors.
- The attempt to insert out-of-range of duplicate key values.
- Permission errors.
- A table does not exist.

Client errors are returned to the client via the MySQL server protocol, where applications are expected to take appropriate action. The `information_schema` \hookrightarrow `.CLIENT_ERRORS_SUMMARY_GLOBAL` table provides a high-level overview, and is useful in the scenario where applications are not correctly handling (or logging) errors returned by the TiDB server.

The summarized counts can be reset with the statement `FLUSH CLIENT_ERRORS_SUMMARY` \hookrightarrow . The summary is local to each TiDB server and is only retained in memory. Summaries will be lost if the TiDB server restarts.

```
USE information_schema;
DESC CLIENT_ERRORS_SUMMARY_GLOBAL;
```

Field	Type	Null	Key	Default	Extra
ERROR_NUMBER	bigint(64)	NO		NULL	
ERROR_MESSAGE	varchar(1024)	NO		NULL	
ERROR_COUNT	bigint(64)	NO		NULL	
WARNING_COUNT	bigint(64)	NO		NULL	
FIRST_SEEN	timestamp	YES		NULL	
LAST_SEEN	timestamp	YES		NULL	

6 rows in set (0.00 sec)

Field description:

- `ERROR_NUMBER`: The MySQL-compatible error number that was returned.
- `ERROR_MESSAGE`: The error message which matches the error number (in prepared statement form).
- `ERROR_COUNT`: The number of times this error was returned.
- `WARNING_COUNT`: The number of times this warning was returned.
- `FIRST_SEEN`: The first time this error (or warning) was sent.
- `LAST_SEEN`: The most recent time this error (or warning) was sent.

The following example shows a warning being generated when connecting to a local TiDB server. The summary is reset after executing `FLUSH CLIENT_ERRORS_SUMMARY`:

```
SELECT 0/0;
SELECT * FROM CLIENT_ERRORS_SUMMARY_GLOBAL;
```

```
FLUSH CLIENT_ERRORS_SUMMARY;
SELECT * FROM CLIENT_ERRORS_SUMMARY_GLOBAL;
```

```
+-----+
| 0/0 |
+-----+
| NULL |
+-----+
1 row in set, 1 warning (0.00 sec)

+-
→ -----+-----+-----+
→
| ERROR_NUMBER | ERROR_MESSAGE | ERROR_COUNT | WARNING_COUNT | FIRST_SEEN |
→ LAST_SEEN      |
+-
→ -----+-----+-----+
→
|       1365 | Division by 0 |          0 |           1 | 2021-03-18 13:10:51
→ | 2021-03-18 13:10:51 |
+-
→ -----+-----+-----+
→
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Empty set (0.00 sec)
```

11.5.17.2.6 CHARACTER_SETS

The CHARACTER_SETS table provides information about [character sets](#). Currently, TiDB only supports some of the character sets.

```
USE information_schema;
DESC character_sets;
```

Field	Type	Null	Key	Default	Extra
CHARACTER_SET_NAME	varchar(32)	YES		NULL	
DEFAULT_COLLATE_NAME	varchar(32)	YES		NULL	
DESCRIPTION	varchar(60)	YES		NULL	
MAXLEN	bigint(3)	YES		NULL	

```
4 rows in set (0.00 sec)
```

```
SELECT * FROM `character_sets`;
```

CHARACTER_SET_NAME	DEFAULT_COLLATE_NAME	DESCRIPTION	MAXLEN
utf8	utf8_bin	UTF-8 Unicode	3
utf8mb4	utf8mb4_bin	UTF-8 Unicode	4
ascii	ascii_bin	US ASCII	1
latin1	latin1_bin	Latin1	1
binary	binary	binary	1

```
5 rows in set (0.00 sec)
```

The description of columns in the CHARACTER_SETS table is as follows:

- CHARACTER_SET_NAME: The name of the character set.
- DEFAULT_COLLATE_NAME: The default collation name of the character set.
- DESCRIPTION: The description of the character set.
- MAXLEN: The maximum length required to store a character in this character set.

11.5.17.2.7 CLUSTER_CONFIG

You can use the CLUSTER_CONFIG cluster configuration table to get the current configuration of all server components in the cluster. This simplifies the usage over earlier releases of TiDB, where obtaining similar information would require accessing the HTTP API end points of each instance.

```
USE information_schema;
DESC cluster_config;
```

Field	Type	Null	Key	Default	Extra
TYPE	varchar(64)	YES		NULL	
INSTANCE	varchar(64)	YES		NULL	
KEY	varchar(256)	YES		NULL	
VALUE	varchar(128)	YES		NULL	

Field description:

- TYPE: The instance type. The optional values are `tidb`, `pd`, and `tikv`.
- INSTANCE: The service address of the instance.

- KEY: The configuration item name.
- VALUE: The configuration item value.

The following example shows how to query the coprocessor configuration on the TiKV instance using the CLUSTER_CONFIG table:

```
SELECT * FROM cluster_config WHERE type='tikv' AND `key` LIKE 'coprocessor%'
→ ;
```

TYPE	INSTANCE	KEY	VALUE
tikv	127.0.0.1:20165	coprocessor.batch-split-limit	10
tikv	127.0.0.1:20165	coprocessor.region-max-keys	1440000
tikv	127.0.0.1:20165	coprocessor.region-max-size	144MiB
tikv	127.0.0.1:20165	coprocessor.region-split-keys	960000
tikv	127.0.0.1:20165	coprocessor.region-split-size	96MiB
tikv	127.0.0.1:20165	coprocessor.split-region-on-table	false

6 rows in set (0.00 sec)

11.5.17.2.8 CLUSTER_HARDWARE

The CLUSTER_HARDWARE hardware system table provides the hardware information of the server where each instance of the cluster is located.

```
USE information_schema;
DESC cluster_hardware;
```

Field	Type	Null	Key	Default	Extra
TYPE	varchar(64)	YES		NULL	
INSTANCE	varchar(64)	YES		NULL	
DEVICE_TYPE	varchar(64)	YES		NULL	
DEVICE_NAME	varchar(64)	YES		NULL	
NAME	varchar(256)	YES		NULL	
VALUE	varchar(128)	YES		NULL	

6 rows in set (0.00 sec)

Field description:

- TYPE: Corresponds to the TYPE field in the `information_schema.cluster_info` table. The optional values are `tidb`, `pd`, and `tikv`.

- INSTANCE: Corresponds to the INSTANCE field in the `information_schema`.
↪ `cluster_info` cluster information table.
- DEVICE_TYPE: Hardware type. Currently, you can query the `cpu`, `memory`, `disk`, and `net` types.
- DEVICE_NAME: Hardware name. The value of DEVICE_NAME varies with DEVICE_TYPE.
 - `cpu`: The hardware name is `cpu`.
 - `memory`: The hardware name is `memory`.
 - `disk`: The disk name.
 - `net`: The network card name.
- NAME: The different information names of the hardware. For example, `cpu` has two information names: `cpu-logical-cores` and `cpu-physical-cores`, which respectively mean logical core numbers and physical core numbers.
- VALUE: The value of the corresponding hardware information, such as the disk volume and CPU core numbers.

The following example shows how to query the CPU information using the `CLUSTER_HARDWARE` table:

```
SELECT * FROM cluster.hardware WHERE device_type='cpu' AND device_name='cpu'
    ↪ AND name LIKE '%cores';
```

TYPE	INSTANCE	DEVICE_TYPE	DEVICE_NAME	NAME	VALUE
tidb	0.0.0.0:4000	cpu	cpu	cpu-logical-cores	16
tidb	0.0.0.0:4000	cpu	cpu	cpu-physical-cores	8
pd	127.0.0.1:2379	cpu	cpu	cpu-logical-cores	16
pd	127.0.0.1:2379	cpu	cpu	cpu-physical-cores	8
tikv	127.0.0.1:20165	cpu	cpu	cpu-logical-cores	16
tikv	127.0.0.1:20165	cpu	cpu	cpu-physical-cores	8

6 rows in set (0.03 sec)

11.5.17.2.9 CLUSTER_INFO

The `CLUSTER_INFO` cluster topology table provides the current topology information of the cluster, the version information of each instance, the Git Hash corresponding to the instance version, the starting time of each instance, and the running time of each instance.

```
USE information_schema;
desc cluster_info;
```

Field	Type	Null	Key	Default	Extra
TYPE	varchar(64)	YES		NULL	
INSTANCE	varchar(64)	YES		NULL	
STATUS_ADDRESS	varchar(64)	YES		NULL	
VERSION	varchar(64)	YES		NULL	
GIT_HASH	varchar(64)	YES		NULL	
START_TIME	varchar(32)	YES		NULL	
UPTIME	varchar(32)	YES		NULL	

7 rows in set (0.00 sec)

Field description:

- **TYPE**: The instance type. The optional values are `tidb`, `pd`, and `tikv`.
- **INSTANCE**: The instance address, which is a string in the format of IP:PORT.
- **STATUS_ADDRESS**: The service address of HTTP API. Some commands in tikv-ctl, pd-ctl, or tidb-ctl might use this API and this address. You can also get more cluster information via this address. Refer to [TiDB HTTP API document](#) for details.
- **VERSION**: The semantic version number of the corresponding instance. To be compatible with the MySQL version number, the TiDB version is displayed in the format of `#{mysql-version}-#{tidb-version}`.
- **GIT_HASH**: The Git Commit Hash when compiling the instance version, which is used to identify whether two instances are of the absolutely consistent version.
- **START_TIME**: The starting time of the corresponding instance.
- **UPTIME**: The uptime of the corresponding instance.

```
SELECT * FROM cluster_info;
```

TYPE	INSTANCE	STATUS_ADDRESS	VERSION	GIT_HASH	START_TIME	UPTIME

```

| tidb | 0.0.0.0:4000 | 0.0.0.0:10080 | 4.0.0-beta.2 | 0
  ↳ df3b74f55f8fbde39bbd5d471783f49dc10f7 | 2020-07-05T09:25:53-06:00 |
  ↳ 26h39m4.352862693s |
| pd   | 127.0.0.1:2379 | 127.0.0.1:2379 | 4.1.0-alpha | 1
  ↳ ad59bcbf36d87082c79a1fffa3b0895234ac862 | 2020-07-05T09:25:47-06:00 |
  ↳ 26h39m10.352868103s |
| tikv | 127.0.0.1:20165 | 127.0.0.1:20180 | 4.1.0-alpha | 1
  ↳ b45e052df8fb5d66aa8b3a77b5c992ddbfbb79df | 2020-07-05T09:25:50-06:00 |
  ↳ 26h39m7.352869963s |
+---+
  ↳ +-----+
  ↳
3 rows in set (0.00 sec)

```

11.5.17.2.10 CLUSTER_LOAD

The CLUSTER_LOAD cluster load table provides the current load information of the server where each instance of the TiDB cluster is located.

```
USE information_schema;
DESC cluster_load;
```

Field	Type	Null	Key	Default	Extra
TYPE	varchar(64)	YES		NULL	
INSTANCE	varchar(64)	YES		NULL	
DEVICE_TYPE	varchar(64)	YES		NULL	
DEVICE_NAME	varchar(64)	YES		NULL	
NAME	varchar(256)	YES		NULL	
VALUE	varchar(128)	YES		NULL	

```
6 rows in set (0.00 sec)
```

Field description:

- **TYPE**: Corresponds to the TYPE field in the `information_schema.cluster_info` table. The optional values are `tidb`, `pd`, and `tikv`.
- **INSTANCE**: Corresponds to the INSTANCE field in the `information_schema.cluster_info` cluster information table.
- **DEVICE_TYPE**: Hardware type. Currently, you can query the `cpu`, `memory`, `disk`, and `net` types.
- **DEVICE_NAME**: Hardware name. The value of DEVICE_NAME varies with DEVICE_TYPE.
 - `cpu`: The hardware name is `cpu`.

- **disk**: The disk name.
 - **net**: The network card name.
 - **memory**: The hardware name is memory.
- **NAME**: Different load types. For example, cpu has three load types: `load1`, `load5`, and `load15`, which respectively mean the average load of cpu within 1 minute, 5 minutes, and 15 minutes.
 - **VALUE**: The value of the hardware load. For example, `1min`, `5min`, and `15min` respectively mean the average load of the hardware within 1 minute, 5 minutes, and 15 minutes.

The following example shows how to query the current load information of cpu using the `CLUSTER_LOAD` table:

```
SELECT * FROM cluster_load WHERE device_type='cpu' AND device_name='cpu';
```

TYPE	INSTANCE	DEVICE_TYPE	DEVICE_NAME	NAME	VALUE
tidb	0.0.0.0:4000	cpu	cpu	load1	0.13
tidb	0.0.0.0:4000	cpu	cpu	load5	0.25
tidb	0.0.0.0:4000	cpu	cpu	load15	0.31
pd	127.0.0.1:2379	cpu	cpu	load1	0.13
pd	127.0.0.1:2379	cpu	cpu	load5	0.25
pd	127.0.0.1:2379	cpu	cpu	load15	0.31
tikv	127.0.0.1:20165	cpu	cpu	load1	0.13
tikv	127.0.0.1:20165	cpu	cpu	load5	0.25
tikv	127.0.0.1:20165	cpu	cpu	load15	0.31

9 rows in set (1.50 sec)

11.5.17.2.11 CLUSTER_LOG

You can query cluster logs on the `CLUSTER_LOG` cluster log table. By pushing down query conditions to each instance, the impact of the query on cluster performance is less than that of the `grep` command.

To get the logs of the TiDB cluster before v4.0, you need to log in to each instance to summarize logs. This cluster log table in 4.0 provides the global and time-ordered log search result, which makes it easier to track full-link events. For example, by searching logs according to the `region_id`, you can query all logs in the life cycle of this Region. Similarly, by searching the full link log through the slow log's `txn_id`, you can query the flow and the number of keys scanned by this transaction at each instance.

```
USE information_schema;
DESC cluster_log;
```

Field	Type	Null	Key	Default	Extra
TIME	varchar(32)	YES		NULL	
TYPE	varchar(64)	YES		NULL	
INSTANCE	varchar(64)	YES		NULL	
LEVEL	varchar(8)	YES		NULL	
MESSAGE	var_string(1024)	YES		NULL	

5 rows in set (0.00 sec)

Field description:

- TIME: The time to print the log.
- TYPE: The instance type. The optional values are `tidb`, `pd`, and `tikv`.
- INSTANCE: The service address of the instance.
- LEVEL: The log level.
- MESSAGE: The log content.

Note:

- All fields of the cluster log table are pushed down to the corresponding instance for execution. To reduce the overhead of using the cluster log table, you must specify the keywords used for the search, the time range, and as many conditions as possible. For example, `select * from cluster_log where message like '%ddl' and time > '2020-05-18 20:40:00' and time<'2020-05-18 21:40:00' and type='tidb'`.
- The `message` field supports the `like` and `regexp` regular expressions, and the corresponding pattern is encoded as `regexp`. Specifying multiple `message` conditions is equivalent to the pipeline form of the `grep` command. For example, executing the `select * from cluster_log where message like 'coprocessor%' and message regexp '.*slow.*'` statement is equivalent to executing `grep 'coprocessor' xxx.log | grep -E '.*slow.*'` on all cluster instances.

The following example shows how to query the execution process of a DDL statement using the `CLUSTER_LOG` table:

```
SELECT time,instance,left(message,150) FROM cluster_log WHERE message LIKE
    ↪ '%ddl%job%ID:80%' AND type='tidb' AND time > '2020-05-18 20:40:00'
    ↪ AND time < '2020-05-18 21:40:00'
```

time	instance	left(message,150)
2020/05/18 21:37:54.784	127.0.0.1:4002	[ddl_worker.go:261] ["[ddl] add DDL jobs"] ["batch count"=1] [jobs="ID:80, Type:create table, State: none, SchemaState:none, SchemaID:1, TableID:79, Ro
2020/05/18 21:37:54.784	127.0.0.1:4002	[ddl.go:477] ["[ddl] start DDL job"] [job="ID:80, Type:create table, State:none, SchemaState:none, SchemaID:1, TableID:79, RowCount:0, ArgLen:1, start
2020/05/18 21:37:55.327	127.0.0.1:4000	[ddl_worker.go:568] ["[ddl] run DDL job"] [worker="worker 1, tp general"] [job="ID:80, Type:create table, State:none, SchemaState:none, SchemaID:1, Ta
2020/05/18 21:37:55.381	127.0.0.1:4000	[ddl_worker.go:763] ["[ddl] wait latest schema version changed"] [worker="worker 1, tp general"] [ver=70] ["take time"=50.809848ms] [job="ID:80, Type:
2020/05/18 21:37:55.382	127.0.0.1:4000	[ddl_worker.go:359] ["[ddl] finish DDL job"] [worker="worker 1, tp general"] [job="ID:80, Type: create table, State:synced, SchemaState:public, Schema
2020/05/18 21:37:55.786	127.0.0.1:4002	[ddl.go:509] ["[ddl] DDL job is finished"] [jobID=80]

The query results above show the process of executing a DDL statement:

1. The request with a DDL JOB ID of 80 is sent to the 127.0.0.1:4002 TiDB instance.
2. The 127.0.0.1:4000 TiDB instance processes this DDL request, which indicates that the 127.0.0.1:4000 instance is the DDL owner at that time.
3. The request with a DDL JOB ID of 80 has been processed.

11.5.17.2.12 CLUSTER_SYSTEMINFO

You can use the `CLUSTER_SYSTEMINFO` kernel parameter table to query the kernel configuration information of the server where all instances of the cluster are located. Currently, you can query the information of the `sysctl` system.

```
USE information_schema;
DESC cluster_systeminfo;
```

Field	Type	Null	Key	Default	Extra
TYPE	varchar(64)	YES		NULL	
INSTANCE	varchar(64)	YES		NULL	
SYSTEM_TYPE	varchar(64)	YES		NULL	
SYSTEM_NAME	varchar(64)	YES		NULL	
NAME	varchar(256)	YES		NULL	
VALUE	varchar(128)	YES		NULL	

6 rows in set (0.00 sec)

Field description:

- **TYPE**: Corresponds to the `TYPE` field in the `information_schema.cluster_info` table. The optional values are `tidb`, `pd`, and `tikv`.
- **INSTANCE**: Corresponds to the `INSTANCE` field in the `information_schema.cluster_info` cluster information table.
- **SYSTEM_TYPE**: The system type. Currently, you can query the `system` system type.
- **SYSTEM_NAME**: The system name. Currently, you can query the `sysctl` system name.
- **NAME**: The configuration name corresponding to `sysctl`.
- **VALUE**: The value of the configuration item corresponding to `sysctl`.

The following example shows how to query the kernel version of all servers in the cluster using the `CLUSTER_SYSTEMINFO` system information table.

```
SELECT * FROM cluster_systeminfo WHERE name LIKE '%kernel.osrelease%'
```

TYPE	INSTANCE	SYSTEM_TYPE	SYSTEM_NAME	NAME	VALUE
tidb	172.16.5.40:4008	system	sysctl	kernel.osrelease	3.10.0-862.14.4.el7.x86_64

pd 172.16.5.40:20379 system sysctl kernel.osrelease
↳ 3.10.0-862.14.4.el7.x86_64
tikv 172.16.5.40:21150 system sysctl kernel.osrelease
↳ 3.10.0-862.14.4.el7.x86_64
+--
↳ -----+-----+-----+-----+-----+
↳

11.5.17.2.13 COLLATIONS

The COLLATIONS table provides a list of collations that correspond to character sets in the CHARACTER_SETS table. Currently, this table is included only for compatibility with MySQL.

```
USE information_schema;
DESC collations;
```

Field	Type	Null	Key	Default	Extra
COLLATION_NAME	varchar(32)	YES		NULL	
CHARACTER_SET_NAME	varchar(32)	YES		NULL	
ID	bigint(11)	YES		NULL	
IS_DEFAULT	varchar(3)	YES		NULL	
IS_COMPILED	varchar(3)	YES		NULL	
SORTLEN	bigint(3)	YES		NULL	

6 rows in set (0.00 sec)

```
SELECT * FROM collations WHERE character_set_name='utf8mb4';
```

+--
↳ -----+-----+-----+-----+-----+
↳
COLLATION_NAME CHARACTER_SET_NAME ID IS_DEFAULT IS_COMPILED
↳ SORTLEN
+--
↳ -----+-----+-----+-----+-----+
↳
utf8mb4_bin utf8mb4 46 Yes Yes 1
+--
↳ -----+-----+-----+-----+-----+
↳

1 row in set (0.00 sec)

The description of columns in the `COLLATION` table is as follows:

- `COLLATION_NAME`: The name of the collation.
- `CHARACTER_SET_NAME`: The name of the character set which the collation belongs to.
- `ID`: The ID of the collation.
- `IS_DEFAULT`: Whether this collation is the default collation of the character set it belongs to.
- `IS_COMPILED`: Whether the character set is compiled into the server.
- `SORTLEN`: The minimum length of memory allocated when the collation sorts characters.

11.5.17.2.14 COLLATION_CHARACTER_SET_APPLICABILITY

The `COLLATION_CHARACTER_SET_APPLICABILITY` table maps collations to the applicable character set name. Similar to the `COLLATIONS` table, it is included only for compatibility with MySQL.

```
USE information_schema;
DESC collation_character_set_applicability;
```

Field	Type	Null	Key	Default	Extra
<code>COLLATION_NAME</code>	<code>varchar(32)</code>	<code>NO</code>		<code>NULL</code>	
<code>CHARACTER_SET_NAME</code>	<code>varchar(32)</code>	<code>NO</code>		<code>NULL</code>	

2 rows in set (0.00 sec)

```
SELECT * FROM collation_character_set_applicability WHERE
  ↪ character_set_name='utf8mb4';
```

<code>COLLATION_NAME</code>	<code>CHARACTER_SET_NAME</code>
<code>utf8mb4_bin</code>	<code>utf8mb4</code>

1 row in set (0.00 sec)

The description of columns in the `COLLATION_CHARACTER_SET_APPLICABILITY` table is as follows:

- `COLLATION_NAME`: The name of the collation.
- `CHARACTER_SET_NAME`: The name of the character set which the collation belongs to.

11.5.17.2.15 COLUMNS

The COLUMNS table provides detailed information about columns in tables.

```
USE information_schema;
DESC columns;
```

Field	Type	Null	Key	Default	Extra
TABLE_CATALOG	varchar(512)	YES		NULL	
TABLE_SCHEMA	varchar(64)	YES		NULL	
TABLE_NAME	varchar(64)	YES		NULL	
COLUMN_NAME	varchar(64)	YES		NULL	
ORDINAL_POSITION	bigint(64)	YES		NULL	
COLUMN_DEFAULT	text	YES		NULL	
IS_NULLABLE	varchar(3)	YES		NULL	
DATA_TYPE	varchar(64)	YES		NULL	
CHARACTER_MAXIMUM_LENGTH	bigint(21)	YES		NULL	
CHARACTER_OCTET_LENGTH	bigint(21)	YES		NULL	
NUMERIC_PRECISION	bigint(21)	YES		NULL	
NUMERIC_SCALE	bigint(21)	YES		NULL	
DATETIME_PRECISION	bigint(21)	YES		NULL	
CHARACTER_SET_NAME	varchar(32)	YES		NULL	
COLLATION_NAME	varchar(32)	YES		NULL	
COLUMN_TYPE	text	YES		NULL	
COLUMN_KEY	varchar(3)	YES		NULL	
EXTRA	varchar(30)	YES		NULL	
PRIVILEGES	varchar(80)	YES		NULL	
COLUMN_COMMENT	varchar(1024)	YES		NULL	
GENERATION_EXPRESSION	text	NO		NULL	

```
21 rows in set (0.00 sec)
```

```
CREATE TABLE test.t1 (a int);
SELECT * FROM columns WHERE table_schema='test' AND TABLE_NAME='t1'\G
```

```
***** 1. row *****
    TABLE_CATALOG: def
    TABLE_SCHEMA: test
```

```

    TABLE_NAME: t1
    COLUMN_NAME: a
    ORDINAL_POSITION: 1
    COLUMN_DEFAULT: NULL
    IS_NULLABLE: YES
    DATA_TYPE: int
CHARACTER_MAXIMUM_LENGTH: NULL
CHARACTER_OCTET_LENGTH: NULL
    NUMERIC_PRECISION: 11
    NUMERIC_SCALE: 0
DATETIME_PRECISION: NULL
CHARACTER_SET_NAME: NULL
    COLLATION_NAME: NULL
    COLUMN_TYPE: int(11)
    COLUMN_KEY:
    EXTRA:
    PRIVILEGES: select,insert,update,references
    COLUMN_COMMENT:
GENERATION_EXPRESSION:
1 row in set (0.02 sec)

```

The description of columns in the COLUMNS table is as follows:

- **TABLE_CATALOG**: The name of the catalog to which the table with the column belongs. The value is always `def`.
- **TABLE_SCHEMA**: The name of the schema in which the table with the column is located.
- **TABLE_NAME**: The name of the table with the column.
- **COLUMN_NAME**: The name of the column.
- **ORDINAL_POSITION**: The position of the column in the table.
- **COLUMN_DEFAULT**: The default value of the column. If the explicit default value is `NULL`, or if the column definition does not include the `default` clause, this value is `NULL`.
- **IS_NULLABLE**: Whether the column is nullable. If the column can store null values, this value is `YES`; otherwise, it is `NO`.
- **DATA_TYPE**: The type of data in the column.
- **CHARACTER_MAXIMUM_LENGTH**: For string columns, the maximum length in characters.
- **CHARACTER_OCTET_LENGTH**: For string columns, the maximum length in bytes.
- **NUMERIC_PRECISION**: The numeric precision of a number-type column.
- **NUMERIC_SCALE**: The numeric scale of a number-type column.
- **DATETIME_PRECISION**: For time-type columns, the fractional seconds precision.
- **CHARACTER_SET_NAME**: The name of the character set of a string column.
- **COLLATION_NAME**: The name of the collation of a string column.
- **COLUMN_TYPE**: The column type.
- **COLUMN_KEY**: Whether this column is indexed. This field might have the following values:

- Empty: This column is not indexed, or this column is indexed and is the second column in a multi-column non-unique index.
- PRI: This column is the primary key or one of multiple primary keys.
- UNI: This column is the first column of the unique index.
- MUL: The column is the first column of a non-unique index, in which a given value is allowed to occur for multiple times.
- EXTRA: Any additional information of the given column.
- PRIVILEGES: The privilege that the current user has on this column. Currently, this value is fixed in TiDB, and is always `select,insert,update,references`.
- COLUMN_COMMENT: Comments contained in the column definition.
- GENERATION_EXPRESSION: For generated columns, this value displays the expression used to calculate the column value. For non-generated columns, the value is empty.

The corresponding SHOW statement is as follows:

```
SHOW COLUMNS FROM t1 FROM test;
```

Field	Type	Null	Key	Default	Extra
a	int(11)	YES		NULL	

1 row in set (0.00 sec)

11.5.17.2.16 DATA_LOCK_WAITS

The `DATA_LOCK_WAITS` table shows the ongoing pessimistic locks waiting on all TiKV nodes in the cluster.

```
USE information_schema;
DESC data_lock_waits;
```

Field	Type	Null	Key	Default	Extra
<code>KEY</code>	text	NO		NULL	
<code>KEY_INFO</code>	text	YES		NULL	
<code>TRX_ID</code>	bigint(21) unsigned	NO		NULL	
<code>CURRENT_HOLDING_TRX_ID</code>	bigint(21) unsigned	NO		NULL	
<code>SQL_DIGEST</code>	varchar(64)	YES		NULL	
<code>SQL_DIGEST_TEXT</code>	text	YES		NULL	

+--	→	-----+-----+-----+-----+
	→	

The meaning of each column field in the `DATA_LOCK_WAIT` table is as follows:

- `KEY`: The key that is waiting for the lock and in the hexadecimal form.
- `KEY_INFO`: The detailed information of `KEY`. See the [KEY_INFO](#) section.
- `TRX_ID`: The ID of the transaction that is waiting for the lock. This ID is also the `start_ts` of the transaction.
- `CURRENT_HOLDING_TRX_ID`: The ID of the transaction that currently holds the lock. This ID is also the `start_ts` of the transaction.
- `SQL_DIGEST`: The digest of the SQL statement that is currently blocked in the lock-waiting transaction.
- `SQL_DIGEST_TEXT`: The normalized SQL statement (the SQL statement without arguments and formats) that is currently blocked in the lock-waiting transaction. It corresponds to `SQL_DIGEST`.

Warning:

- Only the users with the `PROCESS` privilege can query this table.
- Currently, the table can only record the **pessimistic lock waiting** information. If an optimistic transaction (such as an autocommit transaction) is blocked by a pessimistic lock, this table will not display the corresponding lock waiting information.
- The information in the `DATA_LOCK_WAIT` table is obtained in real time from all TiKV nodes during the query. Currently, even if a query has the `WHERE` condition, the information collection is still performed on all TiKV nodes. If your cluster is large and the load is high, querying this table might cause potential risk of performance jitter. Therefore, use it according to your actual situation.
- Information from different TiKV nodes is NOT guaranteed to be snapshots of the same time.
- The information (SQL digest) in the `SQL_DIGEST` column is the hash value calculated from the normalized SQL statement. The information in the `SQL_DIGEST_TEXT` column is internally queried from statements summary tables, so it is possible that the corresponding statement cannot be found internally. For the detailed description of SQL digests and the statements summary tables, see [Statement Summary Tables](#).

`KEY_INFO`

The `KEY_INFO` column shows the detailed information of the `KEY` column. The information is shown in the JSON format. The description of each field is as follows:

- `"db_id"`: The ID of the schema to which the key belongs.
- `"db_name"`: The name of the schema to which the key belongs.
- `"table_id"`: The ID of the table to which the key belongs.
- `"table_name"`: The name of the table to which the key belongs.
- `"partition_id"`: The ID of the partition where the key is located.
- `"partition_name"`: The name of the partition where the key is located.
- `"handle_type"`: The handle type of the row key (that is, the key that stores a row of data). The possible values are as follows:
 - `"int"`: The handle type is int, which means that the handle is the row ID.
 - `"common"`: The handle type is not int64. This type is shown in the non-int primary key when clustered index is enabled.
 - `"unknown"`: The handle type is currently not supported.
- `"handle_value"`: The handle value.
- `"index_id"`: The index ID to which the index key (the key that stores the index) belongs.
- `"index_name"`: The name of the index to which the index key belongs.
- `"index_values"`: The index value in the index key.

In the above fields, if the information of a field is not applicable or currently unavailable, the field is omitted in the query result. For example, the row key information does not contain `index_id`, `index_name`, and `index_values`; the index key does not contain `handle_type` and `handle_value`; non-partitioned tables do not display `partition_id` and `partition_name` ; the key information in the deleted table cannot obtain schema information such as `table_name`, `db_id`, `db_name`, and `index_name`, and it is unable to distinguish whether the table is a partitioned table.

Note:

If a key comes from a table with partitioning enabled, and the information of the schema to which the key belongs cannot be queried due to some reasons (for example, the table to which the key belongs has been deleted) during the query, the ID of the partition to which the key belongs might appear in the `table_id` field. This is because TiDB encodes the keys of different partitions in the same way as it encodes the keys of several independent tables. Therefore, when the schema information is missing, TiDB cannot confirm whether the key belongs to an unpartitioned table or to one partition of a table.

Example

```
select * from information_schema.data_lock_waits\G
```

11.5.17.2.17 DDL JOBS

The `DDL_JOBS` table provides an `INFORMATION_SCHEMA` interface to the `ADMIN SHOW DDL JOBS` command. It provides both the current status and a short history of DDL operations across the TiDB cluster.

```
USE information_schema;  
DESC ddl_jobs;
```

Field	Type	Null	Key	Default	Extra
JOB_ID	bigint(21)	YES		NULL	
DB_NAME	varchar(64)	YES		NULL	
TABLE_NAME	varchar(64)	YES		NULL	
JOB_TYPE	varchar(64)	YES		NULL	
SCHEMA_STATE	varchar(64)	YES		NULL	
SCHEMA_ID	bigint(21)	YES		NULL	
TABLE_ID	bigint(21)	YES		NULL	
ROW_COUNT	bigint(21)	YES		NULL	
START_TIME	datetime	YES		NULL	
END_TIME	datetime	YES		NULL	
STATE	varchar(64)	YES		NULL	
QUERY	varchar(64)	YES		NULL	

```
12 rows in set (0.00 sec)
```

```
SELECT * FROM ddl_jobs LIMIT 3\G
```

```
***** 1. row *****
  JOB_ID: 44
  DB_NAME: mysql
  TABLE_NAME: opt_rule_blacklist
  JOB_TYPE: create table
SCHEMA_STATE: public
  SCHEMA_ID: 3
  TABLE_ID: 43
  ROW_COUNT: 0
START_TIME: 2020-07-06 15:24:27
  END_TIME: 2020-07-06 15:24:27
  STATE: synced
  QUERY: CREATE TABLE IF NOT EXISTS mysql.opt_rule_blacklist (
    name char(100) NOT NULL
);
***** 2. row *****
  JOB_ID: 42
  DB_NAME: mysql
  TABLE_NAME: expr_pushdown_blacklist
  JOB_TYPE: create table
SCHEMA_STATE: public
  SCHEMA_ID: 3
  TABLE_ID: 41
  ROW_COUNT: 0
START_TIME: 2020-07-06 15:24:27
  END_TIME: 2020-07-06 15:24:27
  STATE: synced
  QUERY: CREATE TABLE IF NOT EXISTS mysql.expr_pushdown_blacklist (
    name char(100) NOT NULL,
    store_type char(100) NOT NULL DEFAULT 'tikv,tiflash,tidb',
    reason varchar(200)
);
***** 3. row *****
  JOB_ID: 40
  DB_NAME: mysql
  TABLE_NAME: stats_top_n
  JOB_TYPE: create table
SCHEMA_STATE: public
  SCHEMA_ID: 3
  TABLE_ID: 39
  ROW_COUNT: 0
```

```

START_TIME: 2020-07-06 15:24:26
END_TIME: 2020-07-06 15:24:27
STATE: synced
QUERY: CREATE TABLE if not exists mysql.stats_top_n (
    table_id bigint(64) NOT NULL,
    is_index tinyint(2) NOT NULL,
    hist_id bigint(64) NOT NULL,
    value longblob,
    count bigint(64) UNSIGNED NOT NULL,
    index tbl(table_id, is_index, hist_id)
);
3 rows in set (0.01 sec)

```

11.5.17.2.18 DEADLOCKS

The DEADLOCKS table shows the information of the several deadlock errors that have occurred recently on the current TiDB node.

```

USE information_schema;
DESC deadlocks;

```

Field	Type	Null	Key	Default	Extra
DEADLOCK_ID	bigint(21)	NO		NULL	
OCCUR_TIME	timestamp(6)	YES		NULL	
RETRYABLE	tinyint(1)	NO		NULL	
TRY_LOCK_TRX_ID	bigint(21) unsigned	NO		NULL	
CURRENT_SQL_DIGEST	varchar(64)	YES		NULL	
CURRENT_SQL_DIGEST_TEXT	text	YES		NULL	
KEY	text	YES		NULL	
KEY_INFO	text	YES		NULL	
TRX_HOLDING_LOCK	bigint(21) unsigned	NO		NULL	

The DEADLOCKS table uses multiple rows to show the same deadlock event, and each row displays the information about one of the transactions involved in the deadlock event. If the TiDB node records multiple deadlock errors, each error is distinguished using the DEADLOCK_ID column. The same DEADLOCK_ID indicates the same deadlock event. Note that

`DEADLOCK_ID` does not guarantee global uniqueness and will not be persisted. It only shows the same deadlock event in the same result set.

The meaning of each column field in the `DEADLOCKS` table is as follows:

- `DEADLOCK_ID`: The ID of the deadlock event. When multiple deadlock errors exist in the table, you can use this column to distinguish rows that belong to different deadlock errors.
- `OCCUR_TIME`: The time when the deadlock error occurs.
- `RETRYABLE`: Whether the deadlock error can be retried. For the description of retryable deadlock errors, see the [Retryable deadlock errors](#) section.
- `TRY_LOCK_TRX_ID`: The ID of the transaction that tries to acquire lock. This ID is also the `start_ts` of the transaction.
- `CURRENT_SQL_DIGEST`: The digest of the SQL statement currently being executed in the lock-acquiring transaction.
- `CURRENT_SQL_DIGEST_TEXT`: The normalized form of the SQL statement that is currently being executed in the lock-acquiring transaction.
- `KEY`: The blocked key that the transaction tries to lock. The value of this field is displayed in the form of hexadecimal string.
- `KEY_INFO`: The detailed information of `KEY`. See the [KEY_INFO](#) section.
- `TRX_HOLDING_LOCK`: The ID of the transaction that currently holds the lock on the key and causes blocking. This ID is also the `start_ts` of the transaction.

To adjust the maximum number of deadlock events that can be recorded in the `DEADLOCKS` table, adjust the `pessimistic-txn.deadlock-history-capacity` configuration in the TiDB configuration file. By default, the information of the recent 10 deadlock events is recorded in the table.

Warning:

- Only users with the `PROCESS` privilege can query this table.
- The information (SQL digest) in the `CURRENT_SQL_DIGEST` column is the hash value calculated from the normalized SQL statement. The information in the `CURRENT_SQL_DIGEST_TEXT` column is internally queried from statements summary tables, so it is possible that the corresponding statement cannot be found internally. For the detailed description of SQL digests and the statements summary tables, see [Statement Summary Tables](#).

KEY_INFO

The `KEY_INFO` column shows the detailed information of the `KEY` column. The information is shown in the JSON format. The description of each field is as follows:

- "db_id": The ID of the schema to which the key belongs.
- "db_name": The name of the schema to which the key belongs.
- "table_id": The ID of the table to which the key belongs.
- "table_name": The name of the table to which the key belongs.
- "partition_id": The ID of the partition where the key is located.
- "partition_name": The name of the partition where the key is located.
- "handle_type": The handle type of the row key (that is, the key that stores a row of data). The possible values are as follows:
 - "int": The handle type is int, which means that the handle is the row ID.
 - "common": The handle type is not int64. This type is shown in the non-int primary key when clustered index is enabled.
 - "unknown": The handle type is currently not supported.
- "handle_value": The handle value.
- "index_id": The index ID to which the index key (the key that stores the index) belongs.
- "index_name": The name of the index to which the index key belongs.
- "index_values": The index value in the index key.

In the above fields, if the information of a field is not applicable or currently unavailable, the field is omitted in the query result. For example, the row key information does not contain `index_id`, `index_name`, and `index_values`; the index key does not contain `handle_type` and `handle_value`; non-partitioned tables do not display `partition_id` and `partition_name` → ; the key information in the deleted table cannot obtain schema information such as `table_name`, `db_id`, `db_name`, and `index_name`, and it is unable to distinguish whether the table is a partitioned table.

Note:

If a key comes from a table with partitioning enabled, and the information of the schema to which the key belongs cannot be queried due to some reasons (for example, the table to which the key belongs has been deleted) during the query, the ID of the partition to which the key belongs might appear in the `table_id` field. This is because TiDB encodes the keys of different partitions in the same way as it encodes the keys of several independent tables. Therefore, when the schema information is missing, TiDB cannot confirm whether the key belongs to an unpartitioned table or to one partition of a table.

Retryable deadlock errors

Note:

The DEADLOCKS table does not collect the information of retryable deadlock errors by default. If you want the table to collect the retryable deadlock error information, you can adjust the value of `pessimistic-txn.deadlock→ history-collect-retryable` in the TiDB configuration file.

When transaction A is blocked by a lock already held by transaction B, and transaction B is directly or indirectly blocked by the lock held by the current transaction A, a deadlock error will occur. In this deadlock, there might be two cases:

- Case 1: Transaction B might be (directly or indirectly) blocked by a lock generated by a statement that has been executed after transaction A starts and before transaction A gets blocked.
- Case 2: Transaction B might also be blocked by the statement currently being executed in transaction A.

In case 1, TiDB will report a deadlock error to the client of transaction A and terminate the transaction.

In case 2, the statement currently being executed in transaction A will be automatically retried in TiDB. For example, suppose that transaction A executes the following statement:

```
update t set v = v + 1 where id = 1 or id = 2;
```

Transaction B executes the following two statements successively.

```
update t set v = 4 where id = 2;
update t set v = 2 where id = 1;
```

Then if transaction A locks the two rows with `id = 1` and `id = 2`, and the two transactions run in the following sequence:

1. Transaction A locks the row with `id = 1`.
2. Transaction B executes the first statement and locks the row with `id = 2`.
3. Transaction B executes the second statement and tries to lock the row with `id = 1`, which is blocked by transaction A.
4. Transaction A tries to lock the row with `id = 2` and is blocked by transaction B, which forms a deadlock.

For this case, because the statement of transaction A that blocks other transactions is also the statement currently being executed, the pessimistic lock on the current statement

can be resolved (so that transaction B can continue to run), and the current statement can be retried. TiDB uses the key hash internally to determine whether this is the case.

When a retryable deadlock occurs, the internal automatic retry will not cause a transaction error, so it is transparent to the client. However, if this situation occurs frequently, the performance might be affected. When this occurs, you can see `single statement → deadlock, retry statement` in the TiDB log.

Example 1

Assume that the table definition and the initial data are as follows:

```
create table t (id int primary key, v int);
insert into t values (1, 10), (2, 20);
```

Two transactions are executed in the following order:

Transaction 1	Transaction 2	Description
update t set v = 11 where id = 1;	update t set v = 21 where id = 2;	
update t set v = 12 where id = 2;	update t set v = 22 where id = 1;	Transaction 1 gets blocked Transaction 2 reports conflict

Next, transaction 2 reports a deadlock error. At this time, query the DEADLOCKS table:

```
select * from information_schema.deadlocks;
```

The expected output is as follows:

```
+--+
| 1 | 2021-08-05 11:09:03.230341 | 0 | 426812829645406217 |
|   ↳ 22230766411edb40f27a68dadefc63c6c6970d5827f1e5e22fc97be2c4d8350d |
|   ↳ update `t` set `v` = ? where `id` = ? ; | 7480000000000000000355
|   ↳ F7280000000000000001 | {"db_id":1,"db_name":"test","table_id":53,"
|   ↳ table_name":"t","handle_type":"int","handle_value":"1"} |
|   ↳ 426812829645406216 |
+--+
|   ↳ -----+-----+-----+-----+
|   ↳
```

Two rows of data are generated in the DEADLOCKS table. The DEADLOCK_ID field of both rows is 1, which means that the information in both rows belongs to the same deadlock error. The first row shows that on the key of "7480000000000000000355F7280000000000000002" , the transaction of the ID "426812829645406216" is blocked by the transaction of the ID "426812829645406217". The second row shows that on the key of "7480000000000000000355 F7280000000000000001", the transaction of the ID "426812829645406217" is blocked by the transaction of the ID 426812829645406216, which constitutes mutual blocking and forms a deadlock.

Example 2

Assume that you query the DEADLOCKS table and get the following result:

```
+--+
|   ↳ -----+-----+-----+-----+
|   ↳
```

DEADLOCK_ID	OCCUR_TIME	RETRYABLE	TRY_LOCK_TRX_ID
	CURRENT_SQL_DIGEST		
	CURRENT_SQL_DIGEST_TEXT	KEY	
		KEY_INFO	
	TRX_HOLDING_LOCK		

```
+--+
|   ↳ -----+-----+-----+-----+
|   ↳
```

```
+--+
| 1 | 2021-08-05 11:09:03.230341 | 0 | 426812829645406216 |
|   ↳ 22230766411edb40f27a68dadefc63c6c6970d5827f1e5e22fc97be2c4d8350d |
|   ↳ update `t` set `v` = ? where `id` = ? ; | 7480000000000000000355
|   ↳ F7280000000000000002 | {"db_id":1,"db_name":"test","table_id":53,"
|   ↳ table_name":"t","handle_type":"int","handle_value":"2"} |
|   ↳ 426812829645406217 |
| 1 | 2021-08-05 11:09:03.230341 | 0 | 426812829645406217 |
|   ↳ 22230766411edb40f27a68dadefc63c6c6970d5827f1e5e22fc97be2c4d8350d |
|   ↳ update `t` set `v` = ? where `id` = ? ; | 7480000000000000000355
|   ↳ F7280000000000000001 | {"db_id":1,"db_name":"test","table_id":53,"
|   ↳ table_name":"t","handle_type":"int","handle_value":"1"} |
|   ↳ 426812829645406216 |
```

```

| 2 | 2021-08-05 11:09:21.252154 | 0 | 426812832017809412 |
|   ↳ 22230766411edb40f27a68dадefc63c6c6970d5827f1e5e22fc97be2c4d8350d |
|   ↳ update `t` set `v` = ? where `id` = ? ; | 7480000000000000000355
|   ↳ F7280000000000000002 | {"db_id":1,"db_name":"test","table_id":53,"
|   ↳ table_name":"t","handle_type":"int","handle_value":"2"} |
|   ↳ 426812832017809413 |

| 2 | 2021-08-05 11:09:21.252154 | 0 | 426812832017809413 |
|   ↳ 22230766411edb40f27a68dадefc63c6c6970d5827f1e5e22fc97be2c4d8350d |
|   ↳ update `t` set `v` = ? where `id` = ? ; | 7480000000000000000355
|   ↳ F7280000000000000003 | {"db_id":1,"db_name":"test","table_id":53,"
|   ↳ table_name":"t","handle_type":"int","handle_value":"3"} |
|   ↳ 426812832017809414 |

| 2 | 2021-08-05 11:09:21.252154 | 0 | 426812832017809414 |
|   ↳ 22230766411edb40f27a68dадefc63c6c6970d5827f1e5e22fc97be2c4d8350d |
|   ↳ update `t` set `v` = ? where `id` = ? ; | 7480000000000000000355
|   ↳ F7280000000000000001 | {"db_id":1,"db_name":"test","table_id":53,"
|   ↳ table_name":"t","handle_type":"int","handle_value":"1"} |
|   ↳ 426812832017809412 |

+--+
|   ↳ -----
|   ↳ -----
|   ↳

```

The DEADLOCK_ID column in the above query result shows that the first two rows together represent the information of a deadlock error, and the two transactions that wait for each other form the deadlock. The next three rows together represent the information of another deadlock error, and the three transactions that wait in a cycle form the deadlock.

CLUSTER_DEADLOCKS

The CLUSTER_DEADLOCKS table returns information about the recent deadlock errors on each TiDB node in the entire cluster, which is the combined information of the DEADLOCKS table on each node. CLUSTER_DEADLOCKS also includes an additional INSTANCE column to display the IP address and port of the node to distinguish between different TiDB nodes.

Note that, because DEADLOCK_ID does not guarantee global uniqueness, in the query result of the CLUSTER_DEADLOCKS table, you need to use the INSTANCE and DEADLOCK_ID together to distinguish the information of different deadlock errors in the result set.

```

USE information_schema;
DESC cluster_deadlocks;
```

+--+											
Field		Type		Null	Key	Default	Extra				
+--+											

→						
INSTANCE	varchar(64)	YES	NULL			
DEADLOCK_ID	bigint(21)	NO	NULL			
OCCUR_TIME	timestamp(6)	YES	NULL			
RETRYABLE	tinyint(1)	NO	NULL			
TRY_LOCK_TRX_ID	bigint(21) unsigned	NO	NULL			
CURRENT_SQL_DIGEST	varchar(64)	YES	NULL			
CURRENT_SQL_DIGEST_TEXT	text	YES	NULL			
KEY	text	YES	NULL			
KEY_INFO	text	YES	NULL			
TRX_HOLDING_LOCK	bigint(21) unsigned	NO	NULL			
+--						
→ -----+-----+-----+-----+-----+-----+						
→						

11.5.17.2.19 ENGINES

The ENGINES table provides information about storage engines. For compatibility, TiDB will always describe InnoDB as the only supported engine. In addition, other column values in the ENGINES table are also fixed values.

```
USE information_schema;
DESC engines;
```

Field	Type	Null	Key	Default	Extra
ENGINE	varchar(64)	YES		NULL	
SUPPORT	varchar(8)	YES		NULL	
COMMENT	varchar(80)	YES		NULL	
TRANSACTIONS	varchar(3)	YES		NULL	
XA	varchar(3)	YES		NULL	
SAVEPOINTS	varchar(3)	YES		NULL	

6 rows in set (0.00 sec)

```
SELECT * FROM engines;
```

→						
ENGINE SUPPORT COMMENT						
→ TRANSACTIONS XA SAVEPOINTS						
+-----+-----+-----+-----+-----+-----+-----+						
→						

```
+----+-----+
| InnoDB | DEFAULT | Supports transactions, row-level locking, and foreign
|        |          |    ↢ keys | YES | YES | YES |
+----+-----+
      ↢
1 row in set (0.01 sec)
```

The description of columns in the ENGINES table is as follows:

- **ENGINES**: The name of the storage engine.
- **SUPPORT**: The level of support that the server has on the storage engine. In TiDB, the value is always **DEFAULT**.
- **COMMENT**: The brief comment on the storage engine.
- **TRANSACTIONS**: Whether the storage engine supports transactions.
- **XA**: Whether the storage engine supports XA transactions.
- **SAVEPOINTS**: Whether the storage engine supports **savepoints**.

11.5.17.2.20 INSPECTION_RESULT

TiDB has some built-in diagnostic rules for detecting faults and hidden issues in the system.

The **INSPECTION_RESULT** diagnostic feature can help you quickly find problems and reduce your repetitive manual work. You can use the `select * from information_schema.inspection_result` statement to trigger the internal diagnostics.

The structure of the `information_schema.inspection_result` diagnostic result table `information_schema.inspection_result` is as follows:

```
USE information_schema;
DESC inspection_result;
```

```
+-----+-----+-----+-----+-----+-----+
| Field       | Type           | Null | Key | Default | Extra | 
+-----+-----+-----+-----+-----+-----+
| RULE         | varchar(64)   | YES  |     | NULL    |      | 
| ITEM         | varchar(64)   | YES  |     | NULL    |      | 
| TYPE         | varchar(64)   | YES  |     | NULL    |      | 
| INSTANCE     | varchar(64)   | YES  |     | NULL    |      | 
| STATUS_ADDRESS | varchar(64) | YES  |     | NULL    |      | 
| VALUE         | varchar(64)   | YES  |     | NULL    |      | 
| REFERENCE    | varchar(64)   | YES  |     | NULL    |      | 
| SEVERITY      | varchar(64)   | YES  |     | NULL    |      | 
| DETAILS        | varchar(256)  | YES  |     | NULL    |      | 
+-----+-----+-----+-----+-----+-----+
9 rows in set (0.00 sec)
```

Field description:

- RULE: The name of the diagnostic rule. Currently, the following rules are available:
 - config: Checks whether the configuration is consistent and proper. If the same configuration is inconsistent on different instances, a warning diagnostic result is generated.
 - version: The consistency check of version. If the same version is inconsistent on different instances, a warning diagnostic result is generated.
 - node-load: Checks the server load. If the current system load is too high, the corresponding warning diagnostic result is generated.
 - critical-error: Each module of the system defines critical errors. If a critical error exceeds the threshold within the corresponding time period, a warning diagnostic result is generated.
 - threshold-check: The diagnostic system checks the thresholds of key metrics. If a threshold is exceeded, the corresponding diagnostic information is generated.
- ITEM: Each rule diagnoses different items. This field indicates the specific diagnostic items corresponding to each rule.
- TYPE: The instance type of the diagnostics. The optional values are tidb, pd, and tikv.
- INSTANCE: The specific address of the diagnosed instance.
- STATUS_ADDRESS: The HTTP API service address of the instance.
- VALUE: The value of a specific diagnostic item.
- REFERENCE: The reference value (threshold value) for this diagnostic item. If VALUE exceeds the threshold, the corresponding diagnostic information is generated.
- SEVERITY: The severity level. The optional values are warning and critical.
- DETAILS: Diagnostic details, which might also contain SQL statement(s) or document links for further diagnostics.

Diagnostics example

Diagnose issues currently existing in the cluster.

```
SELECT * FROM information_schema.inspection_result\G
```

```
*****[ 1. row ]*****
RULE | config
ITEM | log.slow-threshold
TYPE | tidb
INSTANCE | 172.16.5.40:4000
VALUE | 0
REFERENCE | not 0
SEVERITY | warning
DETAILS | slow-threshold = 0 will record every query to slow log, it may
         ↪ affect performance
*****[ 2. row ]*****
RULE | version
ITEM | git_hash
```

```

TYPE      | tidb
INSTANCE  |
VALUE    | inconsistent
REFERENCE | consistent
SEVERITY   | critical
DETAILS   | the cluster has 2 different tidb version, execute the sql to see
          ↪ more detail: select * from information_schema.cluster_info where type
          ↪ ='tidb'
*****[ 3. row ]*****
RULE      | threshold-check
ITEM      | storage-write-duration
TYPE      | tikv
INSTANCE  | 172.16.5.40:23151
VALUE    | 130.417
REFERENCE | < 0.100
SEVERITY   | warning
DETAILS   | max duration of 172.16.5.40:23151 tikv storage-write-duration was
          ↪ too slow
*****[ 4. row ]*****
RULE      | threshold-check
ITEM      | rocksdb-write-duration
TYPE      | tikv
INSTANCE  | 172.16.5.40:20151
VALUE    | 108.105
REFERENCE | < 0.100
SEVERITY   | warning
DETAILS   | max duration of 172.16.5.40:20151 tikv rocksdb-write-duration was
          ↪ too slow

```

The following issues can be detected from the diagnostic result above:

- The first row indicates that TiDB's `log.slow-threshold` value is configured to 0, which might affect performance.
- The second row indicates that two different TiDB versions exist in the cluster.
- The third and fourth rows indicate that the TiKV write delay is too long. The expected delay is no more than 0.1 second, while the actual delay is far longer than expected.

You can also diagnose issues existing within a specified range, such as from “2020-03-26 00:03:00” to “2020-03-26 00:08:00”. To specify the time range, use the SQL Hint of `/*+ time_range()*/`. See the following query example:

```
select /*+ time_range("2020-03-26 00:03:00", "2020-03-26 00:08:00") */ *
       ↪ from information_schema.inspection_result\G
```

```
*****[ 1. row ]*****
RULE      | critical-error
ITEM      | server-down
TYPE      | tidb
INSTANCE  | 172.16.5.40:4009
VALUE     |
REFERENCE |
SEVERITY  | critical
DETAILS   | tidb 172.16.5.40:4009 restarted at time '2020/03/26 00:05:45.670'
*****[ 2. row ]*****
RULE      | threshold-check
ITEM      | get-token-duration
TYPE      | tidb
INSTANCE  | 172.16.5.40:10089
VALUE     | 0.234
REFERENCE | < 0.001
SEVERITY  | warning
DETAILS   | max duration of 172.16.5.40:10089 tidb get-token-duration is too
          ↪ slow
```

The following issues can be detected from the diagnostic result above:

- The first row indicates that the 172.16.5.40:4009 TiDB instance is restarted at 2020/03/26 00:05:45.670.
- The second row indicates that the maximum `get-token-duration` time of the 172.16.5.40:10089 TiDB instance is 0.234s, but the expected time is less than 0.001s.

You can also specify conditions, for example, to query the `critical` level diagnostic results:

```
select * from information_schema.inspection_result where severity='critical'
          ↪ ';
```

Query only the diagnostic result of the `critical-error` rule:

```
select * from information_schema.inspection_result where rule='critical-
          ↪ error';
```

Diagnostic rules

The diagnostic module contains a series of rules. These rules compare the results with the thresholds after querying the existing monitoring tables and cluster information tables. If the results exceed the thresholds, the diagnostics of `warning` or `critical` is generated and the corresponding information is provided in the `details` column.

You can query the existing diagnostic rules by querying the `inspection_rules` system table:

```
select * from information_schema.inspection_rules where type='inspection';
```

NAME	TYPE	COMMENT
config	inspection	
version	inspection	
node-load	inspection	
critical-error	inspection	
threshold-check	inspection	

config diagnostic rule

In the config diagnostic rule, the following two diagnostic rules are executed by querying the `CLUSTER_CONFIG` system table:

- Check whether the configuration values of the same component are consistent. Not all configuration items has this consistency check. The allowlist of consistency check is as follows:

```
// The allowlist of the TiDB configuration consistency check
port
status.status-port
host
path
advertise-address
status.status-port
log.file.filename
log.slow-query-file
tmp-storage-path

// The allowlist of the PD configuration consistency check
advertise-client-urls
advertise-peer-urls
client-urls
data-dir
log-file
log.file.filename
metric.job
name
peer-urls
```

```
// The allowlist of the TiKV configuration consistency check
server.addr
server.advertise-addr
server.status-addr
log-file
raftstore.raftdb-path
storage.data-dir
storage.block-cache.capacity
```

- Check whether the values of the following configuration items are as expected.

Component	Configuration item	Expected value
TiDB	log.slow-threshold	larger than 0
TiKV	raftstore.sync-log	true

version diagnostic rule

The `version` diagnostic rule checks whether the version hash of the same component is consistent by querying the `CLUSTER_INFO` system table. See the following example:

```
SELECT * FROM information_schema.inspection_result WHERE rule='version'\G
```

```
*****[ 1. row ]*****
RULE      | version
ITEM      | git_hash
TYPE      | tidb
INSTANCE  |
VALUE     | inconsistent
REFERENCE | consistent
SEVERITY  | critical
DETAILS   | the cluster has 2 different tidb versions, execute the sql to see
          ↳ more detail: SELECT * FROM information_schema.cluster_info WHERE
          ↳ type='tidb'
```

critical-error diagnostic rule

In `critical-error` diagnostic rule, the following two diagnostic rules are executed:

- Detect whether the cluster has the following errors by querying the related monitoring system tables in the metrics schema:

Component	Error	Monitoring
	de-	ip-
	table	tion
TiDB	panic- count	tidb_paniccount_total_count oc- curs in TiDB.
TiDB	binlog- error	tidb_binlog_error_total_count oc- curs when TiDB writes bin- log.
TiKV	critical- error	tikv_critical_error_total_coun criti- cal error of TiKV.
TiKV	schedul- is- busy	tikv_scheduler_is_busy_total_count TiKV sched- uler is too busy, which makes TiKV tem- porar- ily un- avail- able.

Component	Error	Monitoring
	de-	ip-
TiKV	coprocessor_is_busy	The processor_is_busy_total_count is- TiKV busy Co- pro- ces- sor is too busy.
TiKV	channel_full	The channel_full_total_count is- “chan- full nel full” error oc- curs in TiKV.
TiKV	tikv_engine_stall	The tikv_engine_stallte_stall “stall” error oc- curs in TiKV.

- Check whether any component is restarted by querying the `metrics_schema.up` monitoring table and the `CLUSTER_LOG` system table.

threshold-check diagnostic rule

The `threshold-check` diagnostic rule checks whether the following metrics in the cluster exceed the threshold by querying the related monitoring system tables in the metrics schema:

Component	table	value	Description
TiDB	tso- duration	pd_tso<wait_dilation 50ms	MonitorExpected The duration of getting the TSO of trans- ac- tion.
TiDB	get- token- duration	tidb_get_tokenQdlat 1ms	Time it takes to get the to- ken. The re- lated TiDB con- figu- ra- tion item is token → - → limit → .

Component	table	value	Description
TiDB	load-schema-duration	tidb_load_1scheThe_duration	Monitoring met-MonitorExpected
TiKV	scheduler-command-duration	tikv_scheduler_Themand_duration	cmd-0.1s time
TiKV	handle-snapshot-duration	tikv_handle_snTshot_duration	snapshot-30s time

duration it takes for TiKV to up- date the schema meta- data.

duration it takes for TiKV to execute the KV cmd re- quest.

duration it takes for TiKV to han- dle the snap- shot.

Component	table	value	Description
TiKV	storage-tikv_storage_async_request_duration		Monitoring met-MonitorExpected
	write-duration	0.1s	write latency of TiKV.
TiKV	storage-tikv_storage_async_request_duration		
	snapshot-duration	50ms	time it takes for TiKV to get the snapshot.
TiKV	rocksdbtikv_engine_write_duration		
	write-duration	100ms	write latency of TiKV RocksDB.
TiKV	rocksdbtikv_engine_read_get_duration		
	get-duration	50ms	read latency of TiKV RocksDB.
TiKV	rocksdbtikv_engine_read_seek_duration		
	seek-duration	50ms	latency of TiKV RocksDB to execute seek.

Component	table	value	Description
TiKV	schedulertikv_scheduler.Pending_commands	pending- cmd- coun	MonitorExpected number of commands stalled in TiKV.
TiKV	index-block_indexcache_hit	block- cache- hit	The cache hit rate of index block cache in TiKV.
TiKV	tikv_block_filtercache_hit	block- cache- hit	The cache hit rate of filter block cache in TiKV.
TiKV	tikv_block_datacache_hit	block- cache- hit	The cache hit rate of data block cache in TiKV.

Component	table	value	Description
TiKV	leader- score- balance	pd_scheduler_score 0.05	Monitor the leader score of each TiKV instance. The expected difference between instances is less than 5%.

Component	table	value	Description
TiKV	region- score- balance	pd_scheduler_score 0.05	Monitoring met-MonitorExpected the Region score of each TiKV instance is balanced. The expected difference between instances is less than 5%.

Component	Monitoring table	Expected value	Description
TiKV	store- available- balance	pd_scheduler_stores whether the avail- able stor- age of each TiKV in- stance is bal- anced. The ex- pected dif- ference be- tween in- stances is less than 20%.	Monitor the status of each TiKV store to check if it is available and balanced. The expected value for 'available' is 'true' and for 'balance' is 'true'. The description explains that the monitoring table 'pd_scheduler_stores' checks whether the balance of each TiKV instance is balanced. The difference between instances is less than 20%.

Component	table	value	Description
TiKV	region- count	pd_scheduler_st ore count	The number of regions on each TiKV instance. The expected number of regions in a single instance is less than 20,000.

Component	table	value	Description
PD	region-health	pd_region_health	Detects the number of regions that are in the process of scheduling in the cluster. The expected number is less than 100 in total.

In addition, this rule also checks whether the CPU usage of the following threads in a TiKV instance is too high:

- scheduler-worker-cpu
- coprocessor-normal-cpu
- coprocessor-high-cpu
- coprocessor-low-cpu
- grpc-cpu
- raftstore-cpu
- apply-cpu

- storage-readpool-normal-cpu
- storage-readpool-high-cpu
- storage-readpool-low-cpu
- split-check-cpu

The built-in diagnostic rules are constantly being improved. If you have more diagnostic rules, welcome to create a PR or an issue in the [tidb](#) repository.

11.5.17.2.21 INSPECTION_RULES

The `INSPECTION_RULES` table provides information about which diagnostic tests are run in an inspection result. See [inspection result](#) for example usage.

```
USE information_schema;
DESC inspection_rules;
```

Field	Type	Null	Key	Default	Extra
NAME	varchar(64)	YES		NULL	
TYPE	varchar(64)	YES		NULL	
COMMENT	varchar(256)	YES		NULL	

3 rows in set (0.00 sec)

```
SELECT * FROM inspection_rules;
```

NAME	TYPE	COMMENT
config	inspection	
version	inspection	
node-load	inspection	
critical-error	inspection	
threshold-check	inspection	
ddl	summary	
gc	summary	
pd	summary	
query-summary	summary	
raftstore	summary	
read-link	summary	
rocksdb	summary	
stats	summary	
wait-events	summary	
write-link	summary	

```
+-----+-----+-----+
15 rows in set (0.00 sec)
```

11.5.17.2.22 INSPECTION_SUMMARY

In some scenarios, you might need to pay attention only to the monitoring summary of specific links or modules. For example, the number of threads for Coprocessor in the thread pool is configured as 8. If the CPU usage of Coprocessor reaches 750%, you can determine that a risk exists and Coprocessor might become a bottleneck in advance. However, some monitoring metrics vary greatly due to different user workloads, so it is difficult to define specific thresholds. It is important to troubleshoot issues in this scenario, so TiDB provides the `inspection_summary` table for link summary.

The structure of the `information_schema.inspection_summary` inspection summary table is as follows:

```
USE information_schema;
DESC inspection_summary;
```

Field	Type	Null	Key	Default	Extra
RULE	<code>varchar(64)</code>	YES		<code>NULL</code>	
INSTANCE	<code>varchar(64)</code>	YES		<code>NULL</code>	
METRICS_NAME	<code>varchar(64)</code>	YES		<code>NULL</code>	
LABEL	<code>varchar(64)</code>	YES		<code>NULL</code>	
QUANTILE	<code>double</code>	YES		<code>NULL</code>	
AVG_VALUE	<code>double(22,6)</code>	YES		<code>NULL</code>	
MIN_VALUE	<code>double(22,6)</code>	YES		<code>NULL</code>	
MAX_VALUE	<code>double(22,6)</code>	YES		<code>NULL</code>	
COMMENT	<code>varchar(256)</code>	YES		<code>NULL</code>	

```
9 rows in set (0.00 sec)
```

Field description:

- RULE: Summary rules. Because new rules are being added continuously, you can execute the `select * from inspection_rules where type='summary'` statement to query the latest rule list.
- INSTANCE: The monitored instance.
- METRICS_NAME: The monitoring metrics name.
- QUANTILE: Takes effect on monitoring tables that contain QUANTILE. You can specify multiple percentiles by pushing down predicates. For example, you can execute `select * from inspection_summary where rule='ddl' and quantile in (0.80, 0.90, 0.99, 0.999)` to summarize the DDL-related monitoring

metrics and query the P80/P90/P99/P999 results. `AVG_VALUE`, `MIN_VALUE`, and `MAX_VALUE` respectively indicate the average value, minimum value, and maximum value of the aggregation.

- `COMMENT`: The comment about the corresponding monitoring metric.

Note:

Because summarizing all results causes overhead, it is recommended to display the specific rule in the SQL predicate to reduce overhead. For example, executing `select * from inspection_summary where rule in ('→ read-link', 'ddl')` summarizes the read link and DDL-related monitoring metrics.

Usage example:

Both the diagnostic result table and the diagnostic monitoring summary table can specify the diagnostic time range using `hint`. `select /*+ time_range('2020-03-07 → 12:00:00','2020-03-07 13:00:00')*//* from inspection_summary` is the monitoring summary for the 2020-03-07 12:00:00 to 2020-03-07 13:00:00 period. Like the monitoring summary table, you can use the `inspection_summary` table to quickly find the monitoring items with large differences by comparing the data of two different periods.

The following example compares the monitoring metrics of read links in two time periods:

- (2020-01-16 16:00:54.933, 2020-01-16 16:10:54.933)
- (2020-01-16 16:10:54.933, 2020-01-16 16:20:54.933)

```
SELECT
    t1.avg_value / t2.avg_value AS ratio,
    t1.*,
    t2.*
FROM
(
    SELECT
        /*+ time_range("2020-01-16 16:00:54.933", "2020-01-16 16:10:54.933")
        → */ *
    FROM information_schema.inspection_summary WHERE rule='read-link'
) t1
JOIN
(
    SELECT
        /*+ time_range("2020-01-16 16:10:54.933", "2020-01-16 16:20:54.933")
        → */ *
)
```

```

    FROM information_schema.inspection_summary WHERE rule='read-link'
) t2
ON t1.metrics_name = t2.metrics_name
AND t1.instance = t2.instance
AND t1.label = t2.label
ORDER BY
ratio DESC;

```

11.5.17.2.23 KEY_COLUMN_USAGE

The KEY_COLUMN_USAGE table describes the key constraints of the columns, such as the primary key constraint.

```

USE information_schema;
DESC key_column_usage;

```

Field	Type	Null	Key	Default	Extra
CONSTRAINT_CATALOG	varchar(512)	NO		NULL	
CONSTRAINT_SCHEMA	varchar(64)	NO		NULL	
CONSTRAINT_NAME	varchar(64)	NO		NULL	
TABLE_CATALOG	varchar(512)	NO		NULL	
TABLE_SCHEMA	varchar(64)	NO		NULL	
TABLE_NAME	varchar(64)	NO		NULL	
COLUMN_NAME	varchar(64)	NO		NULL	
ORDINAL_POSITION	bigint(10)	NO		NULL	
POSITION_IN_UNIQUE_CONSTRAINT	bigint(10)	YES		NULL	
REFERENCED_TABLE_SCHEMA	varchar(64)	YES		NULL	
REFERENCED_TABLE_NAME	varchar(64)	YES		NULL	
REFERENCED_COLUMN_NAME	varchar(64)	YES		NULL	

12 rows in set (0.00 sec)

```

SELECT * FROM key_column_usage WHERE table_schema='mysql' AND table_name='
    ↪ user';

```

```

***** 1. row *****
CONSTRAINT_CATALOG: def
CONSTRAINT_SCHEMA: mysql
CONSTRAINT_NAME: PRIMARY
TABLE_CATALOG: def

```

```

        TABLE_SCHEMA: mysql
        TABLE_NAME: user
        COLUMN_NAME: Host
        ORDINAL_POSITION: 1
POSITION_IN_UNIQUE_CONSTRAINT: NULL
        REFERENCED_TABLE_SCHEMA: NULL
        REFERENCED_TABLE_NAME: NULL
        REFERENCED_COLUMN_NAME: NULL
***** 2. row *****
        CONSTRAINT_CATALOG: def
        CONSTRAINT_SCHEMA: mysql
        CONSTRAINT_NAME: PRIMARY
        TABLE_CATALOG: def
        TABLE_SCHEMA: mysql
        TABLE_NAME: user
        COLUMN_NAME: User
        ORDINAL_POSITION: 2
POSITION_IN_UNIQUE_CONSTRAINT: NULL
        REFERENCED_TABLE_SCHEMA: NULL
        REFERENCED_TABLE_NAME: NULL
        REFERENCED_COLUMN_NAME: NULL
2 rows in set (0.00 sec)

```

The description of columns in the KEY_COLUMN_USAGE table is as follows:

- **CONSTRAINT_CATALOG**: The name of the catalog to which the constraint belongs. The value is always `def`.
- **CONSTRAINT_SCHEMA**: The name of the schema to which the constraint belongs.
- **CONSTRAINT_NAME**: The name of the constraint.
- **TABLE_CATALOG**: The name of the catalog to which the table belongs. The value is always `def`.
- **TABLE_SCHEMA**: The name of the schema to which the table belongs.
- **TABLE_NAME**: The name of the table with constraints.
- **COLUMN_NAME**: The name of the column with constraints.
- **ORDINAL_POSITION**: The position of the column in the constraint, rather than in the table. The position number starts from 1.
- **POSITION_IN_UNIQUE_CONSTRAINT**: The unique constraint and the primary key constraint are empty. For foreign key constraints, this column is the position of the referenced table's key.
- **REFERENCED_TABLE_SCHEMA**: The name of the schema referenced by the constraint. Currently in TiDB, the value of this column in all constraints is `nil`, except for the foreign key constraint.
- **REFERENCED_TABLE_NAME**: The name of the table referenced by the constraint. Currently in TiDB, the value of this column in all constraints is `nil`, except for the foreign key constraint.

- **REFERENCED_COLUMN_NAME**: The name of the column referenced by the constraint. Currently in TiDB, the value of this column in all constraints is `nil`, except for the foreign key constraint.

11.5.17.2.24 METRICS_SUMMARY

The TiDB cluster has many monitoring metrics. To make it easy to detect abnormal monitoring metrics, TiDB 4.0 introduces the following two monitoring summary tables:

- `information_schema.metrics_summary`
- `information_schema.metrics_summary_by_label`

The two tables summarize all monitoring data for you to check each monitoring metric efficiently. Compared with `information_schema.metrics_summary`, the `information_schema.metrics_summary_by_label` table has an additional `label` column and performs differentiated statistics according to different labels.

```
USE information_schema;
DESC metrics_summary;
```

Field	Type	Null	Key	Default	Extra
METRICS_NAME	varchar(64)	YES		NULL	
QUANTILE	double	YES		NULL	
SUM_VALUE	double(22,6)	YES		NULL	
AVG_VALUE	double(22,6)	YES		NULL	
MIN_VALUE	double(22,6)	YES		NULL	
MAX_VALUE	double(22,6)	YES		NULL	
COMMENT	varchar(256)	YES		NULL	

7 rows in set (0.00 sec)

Field description:

- **METRICS_NAME**: The monitoring table name.
- **QUANTILE**: The percentile. You can specify QUANTILE using SQL statements. For example:
 - `select * from metrics_summary where quantile=0.99` specifies viewing the data of the 0.99 percentile.
 - `select * from metrics_summary where quantile in (0.80, 0.90, 0.99, ↪ 0.999)` specifies viewing the data of the 0.8, 0.90, 0.99, 0.999 percentiles at the same time.

- `SUM_VALUE`, `AVG_VALUE`, `MIN_VALUE`, and `MAX_VALUE` respectively mean the sum, the average value, the minimum value, and the maximum value.
- `COMMENT`: The comment for the corresponding monitoring table.

For example:

To query the three groups of monitoring items with the highest average time consumption in the TiDB cluster within the time range of '`2020-03-08 13:23:00`', '`2020-03-08 13:33:00`', you can directly query the `information_schema.metrics_summary` table and use the `/*+ time_range()*/` hint to specify the time range. The SQL statement is as follows:

```
SELECT /*+ time_range('2020-03-08 13:23:00', '2020-03-08 13:33:00') */ *
FROM information_schema.metrics_summary
WHERE metrics_name LIKE 'tidb%duration'
  AND avg_value > 0
  AND quantile = 0.99
ORDER BY avg_value DESC
LIMIT 3\G
```

```
*****[ 1. row ]*****
METRICS_NAME | tidb_get_token_duration
QUANTILE    | 0.99
SUM_VALUE   | 8.972509
AVG_VALUE   | 0.996945
MIN_VALUE   | 0.996515
MAX_VALUE   | 0.997458
COMMENT     | The quantile of Duration (us) for getting token, it should be
             ↵ small until concurrency limit is reached(second)
*****[ 2. row ]*****
METRICS_NAME | tidb_query_duration
QUANTILE    | 0.99
SUM_VALUE   | 0.269079
AVG_VALUE   | 0.007272
MIN_VALUE   | 0.000667
MAX_VALUE   | 0.01554
COMMENT     | The quantile of TiDB query durations(second)
*****[ 3. row ]*****
METRICS_NAME | tidb_kv_request_duration
QUANTILE    | 0.99
SUM_VALUE   | 0.170232
AVG_VALUE   | 0.004601
MIN_VALUE   | 0.000975
MAX_VALUE   | 0.013
COMMENT     | The quantile of kv requests durations by store
```

Similarly, the following example queries the `metrics_summary_by_label` monitoring summary table:

```
SELECT /*+ time_range('2020-03-08 13:23:00', '2020-03-08 13:33:00') */ *
FROM information_schema.metrics_summary_by_label
WHERE metrics_name LIKE 'tidb%duration'
    AND avg_value > 0
    AND quantile = 0.99
ORDER BY avg_value DESC
LIMIT 10\G
```

```
*****[ 1. row ]*****
INSTANCE | 172.16.5.40:10089
METRICS_NAME | tidb_get_token_duration
LABEL | 
QUANTILE | 0.99
SUM_VALUE | 8.972509
AVG_VALUE | 0.996945
MIN_VALUE | 0.996515
MAX_VALUE | 0.997458
COMMENT | The quantile of Duration (us) for getting token, it should be
        ↪ small until concurrency limit is reached(second)
*****[ 2. row ]*****
INSTANCE | 172.16.5.40:10089
METRICS_NAME | tidb_query_duration
LABEL | Select
QUANTILE | 0.99
SUM_VALUE | 0.072083
AVG_VALUE | 0.008009
MIN_VALUE | 0.007905
MAX_VALUE | 0.008241
COMMENT | The quantile of TiDB query durations(second)
*****[ 3. row ]*****
INSTANCE | 172.16.5.40:10089
METRICS_NAME | tidb_query_duration
LABEL | Rollback
QUANTILE | 0.99
SUM_VALUE | 0.072083
AVG_VALUE | 0.008009
MIN_VALUE | 0.007905
MAX_VALUE | 0.008241
COMMENT | The quantile of TiDB query durations(second)
```

The second and third rows of the query results above indicate that the `Select` and `Rollback` statements on `tidb_query_duration` have a long average execution time.

In addition to the example above, you can use the monitoring summary table to quickly find the module with the largest change from the monitoring data by comparing the full link monitoring items of the two time periods, and quickly locate the bottleneck. The following example compares all monitoring items in two periods (where t1 is the baseline) and sorts these items according to the greatest difference:

- Period t1: ("2020-03-03 17:08:00", "2020-03-03 17:11:00")
- Period t2: ("2020-03-03 17:18:00", "2020-03-03 17:21:00")

The monitoring items of the two time periods are joined according to METRICS_NAME and sorted according to the difference value. TIME_RANGE is the hint that specifies the query time.

```
SELECT GREATEST(t1.avg_value,t2.avg_value)/LEAST(t1.avg_value,
    t2.avg_value) AS ratio,
    t1.metrics_name,
    t1.avg_value as t1_avg_value,
    t2.avg_value as t2_avg_value,
    t2.comment
FROM
    (SELECT /*+ time_range("2020-03-03 17:08:00", "2020-03-03 17:11:00")*/
        *
     FROM information_schema.metrics_summary ) t1
JOIN
    (SELECT /*+ time_range("2020-03-03 17:18:00", "2020-03-03 17:21:00")*/
        *
     FROM information_schema.metrics_summary ) t2
    ON t1.metrics_name = t2.metrics_name
ORDER BY ratio DESC LIMIT 10;
```

+--	ratio	metrics_name	t1_avg_value
+--	t2_avg_value	comment	
→			
→			
5865.59537065 tidb_slow_query_cop_process_total_time 0.016333	95.804724	The total time of TiDB slow query statistics with slow	
		query total cop process time(second)	
3648.74109023 tidb_distsql_partial_scan_key_total_num 10865.666667	39646004.4394	The total num of distsql partial scan key numbers	

267.002351165 tidb_slow_query_cop_wait_total_time 0.003333
↳ 0.890008 The total time of TiDB slow query statistics with
↳ slow query total cop wait time(second)
192.43267836 tikv_cop_total_response_total_size 2515333.66667
↳ 484032394.445
↳
↳
192.43267836 tikv_cop_total_response_size_per_seconds 41922.227778
↳ 8067206.57408
↳
↳
152.780296296 tidb_distsql_scan_key_total_num 5304.333333
↳ 810397.618317 The total num of distsql scan numbers
↳
↳
126.042290167 tidb_distsql_execution_total_time 0.421622
↳ 53.142143 The total time of distsql execution(second)
↳
↳
105.164020657 tikv_cop_scan_details 134.450733
↳ 14139.379665
↳
↳
105.164020657 tikv_cop_scan_details_total 8067.043981
↳ 848362.77991
↳
↳
101.635495394 tikv_cop_scan_keys_num 1070.875
↳ 108838.91113
↳
↳
+--
↳ -----+-----+-----+
↳

From the query result above, you can get the following information:

- **tib_slow_query_cop_process_total_time** (the time consumption of cop process in TiDB slow queries) in the period t2 is 5,865 times higher than that in period t1.
- **tidb_distsql_partial_scan_key_total_num** (the number of keys to scan requested by TiDB's distsql) in period t2 is 3,648 times higher than that in period t1. During period t2, **tidb_slow_query_cop_wait_total_time** (the waiting time of Coprocessor requesting to queue up in the TiDB slow query) is 267 times higher than that in period t1.
- **tikv_cop_total_response_size** (the size of the TiKV Coprocessor request result) in period t2 is 192 times higher than that in period t1.

- `tikv_cop_scan_details` in period t2 (the scan requested by the TiKV Coprocessor) is 105 times higher than that in period t1.

From the result above, you can see that the Coprocessor requests in period t2 are much more than those in period t1. This causes TiKV Coprocessor to be overloaded, and the `cop` task has to wait. It might be that some large queries appear in period t2 that bring more load.

In fact, during the entire time period from t1 to t2, the `go-ycsb` pressure test is running. Then 20 `tpch` queries are running during period t2. So it is the `tpch` queries that cause many Coprocessor requests.

11.5.17.2.25 METRICS_TABLES

The `METRICS_TABLES` table provides the PromQL (Prometheus Query Language) definition for each of the views in the `metrics_schema` database.

```
USE information_schema;
DESC metrics_tables;
```

Field	Type	Null	Key	Default	Extra
TABLE_NAME	varchar(64)	YES		NULL	
PROMQL	varchar(64)	YES		NULL	
LABELS	varchar(64)	YES		NULL	
QUANTILE	double	YES		NULL	
COMMENT	varchar(256)	YES		NULL	

Field description:

- **TABLE_NAME**: Corresponds to the table name in `metrics_schema`.
- **PROMQL**: The working principle of the monitoring table is to map SQL statements to PromQL and convert Prometheus results into SQL query results. This field is the expression template of PromQL. When you query the data of the monitoring table, the query conditions are used to rewrite the variables in this template to generate the final query expression.
- **LABELS**: The label for the monitoring item. Each label corresponds to a column in the monitoring table. If the SQL statement contains the filter of the corresponding column, the corresponding PromQL changes accordingly.
- **QUANTILE**: The percentile. For monitoring data of the histogram type, a default percentile is specified. If the value of this field is 0, it means that the monitoring item corresponding to the monitoring table is not a histogram.
- **COMMENT**: The comment about the monitoring table.

```
SELECT * FROM metrics_tables LIMIT 5\G
```

```
***** 1. row *****
TABLE_NAME: abnormal_stores
  PROMQL: sum(pd_cluster_status{ type=~"store_disconnected_count|"
    ↪ store_unhealth_count|store_low_space_count|store_down_count|
    ↪ store_offline_count|store_tombstone_count"})
  LABELS: instance,type
  QUANTILE: 0
  COMMENT:
***** 2. row *****
TABLE_NAME: etcd_disk_wal_fsync_rate
  PROMQL: delta(etcd_disk_wal_fsync_duration_seconds_count{
    ↪ $LABEL_CONDITIONS}[$RANGE_DURATION])
  LABELS: instance
  QUANTILE: 0
  COMMENT: The rate of writing WAL into the persistent storage
***** 3. row *****
TABLE_NAME: etcd_wal_fsync_duration
  PROMQL: histogram_quantile($QUANTILE, sum(rate(
    ↪ etcd_disk_wal_fsync_duration_seconds_bucket{$LABEL_CONDITIONS} [
    ↪ $RANGE_DURATION])) by (le,instance))
  LABELS: instance
  QUANTILE: 0.99
  COMMENT: The quantile time consumed of writing WAL into the persistent
    ↪ storage
***** 4. row *****
TABLE_NAME: etcd_wal_fsync_total_count
  PROMQL: sum(increase(etcd_disk_wal_fsync_duration_seconds_count{
    ↪ $LABEL_CONDITIONS}[$RANGE_DURATION])) by (instance)
  LABELS: instance
  QUANTILE: 0
  COMMENT: The total count of writing WAL into the persistent storage
***** 5. row *****
TABLE_NAME: etcd_wal_fsync_total_time
  PROMQL: sum(increase(etcd_disk_wal_fsync_duration_seconds_sum{
    ↪ $LABEL_CONDITIONS}[$RANGE_DURATION])) by (instance)
  LABELS: instance
  QUANTILE: 0
  COMMENT: The total time of writing WAL into the persistent storage
5 rows in set (0.00 sec)
```

11.5.17.2.26 PARTITIONS

The PARTITIONS table provides information about partitioned tables.

```
USE information_schema;
DESC partitions;
```

Field	Type	Null	Key	Default	Extra
TABLE_CATALOG	varchar(512)	YES		NULL	
TABLE_SCHEMA	varchar(64)	YES		NULL	
TABLE_NAME	varchar(64)	YES		NULL	
PARTITION_NAME	varchar(64)	YES		NULL	
SUBPARTITION_NAME	varchar(64)	YES		NULL	
PARTITION_ORDINAL_POSITION	bigint(21)	YES		NULL	
SUBPARTITION_ORDINAL_POSITION	bigint(21)	YES		NULL	
PARTITION_METHOD	varchar(18)	YES		NULL	
SUBPARTITION_METHOD	varchar(12)	YES		NULL	
PARTITION_EXPRESSION	longblob	YES		NULL	
SUBPARTITION_EXPRESSION	longblob	YES		NULL	
PARTITION_DESCRIPTION	longblob	YES		NULL	
TABLE_ROWS	bigint(21)	YES		NULL	
AVG_ROW_LENGTH	bigint(21)	YES		NULL	
DATA_LENGTH	bigint(21)	YES		NULL	
MAX_DATA_LENGTH	bigint(21)	YES		NULL	
INDEX_LENGTH	bigint(21)	YES		NULL	
DATA_FREE	bigint(21)	YES		NULL	
CREATE_TIME	datetime	YES		NULL	
UPDATE_TIME	datetime	YES		NULL	
CHECK_TIME	datetime	YES		NULL	
CHECKSUM	bigint(21)	YES		NULL	
PARTITION_COMMENT	varchar(80)	YES		NULL	
NODEGROUP	varchar(12)	YES		NULL	
TABLESPACE_NAME	varchar(64)	YES		NULL	

25 rows in set (0.00 sec)

```
CREATE TABLE test.t1 (id INT NOT NULL PRIMARY KEY) PARTITION BY HASH (id)
    → PARTITIONS 2;
SELECT * FROM partitions WHERE table_schema='test' AND table_name='t1'\G
```

```
***** 1. row *****
TABLE_CATALOG: def
```

```

        TABLE_SCHEMA: test
        TABLE_NAME: t1
        PARTITION_NAME: p0
        SUBPARTITION_NAME: NULL
    PARTITION_ORDINAL_POSITION: 1
SUBPARTITION_ORDINAL_POSITION: NULL
        PARTITION_METHOD: HASH
        SUBPARTITION_METHOD: NULL
        PARTITION_EXPRESSION: `id`
SUBPARTITION_EXPRESSION: NULL
        PARTITION_DESCRIPTION:
            TABLE_ROWS: 0
            AVG_ROW_LENGTH: 0
            DATA_LENGTH: 0
            MAX_DATA_LENGTH: 0
            INDEX_LENGTH: 0
            DATA_FREE: 0
        CREATE_TIME: 2020-07-06 16:35:28
        UPDATE_TIME: NULL
        CHECK_TIME: NULL
        CHECKSUM: NULL
    PARTITION_COMMENT:
        NODEGROUP: NULL
    TABLESPACE_NAME: NULL
***** 2. row *****
        TABLE_CATALOG: def
        TABLE_SCHEMA: test
        TABLE_NAME: t1
        PARTITION_NAME: p1
        SUBPARTITION_NAME: NULL
    PARTITION_ORDINAL_POSITION: 2
SUBPARTITION_ORDINAL_POSITION: NULL
        PARTITION_METHOD: HASH
        SUBPARTITION_METHOD: NULL
        PARTITION_EXPRESSION: `id`
SUBPARTITION_EXPRESSION: NULL
        PARTITION_DESCRIPTION:
            TABLE_ROWS: 0
            AVG_ROW_LENGTH: 0
            DATA_LENGTH: 0
            MAX_DATA_LENGTH: 0
            INDEX_LENGTH: 0
            DATA_FREE: 0
        CREATE_TIME: 2020-07-06 16:35:28
        UPDATE_TIME: NULL

```

```

    CHECK_TIME: NULL
    CHECKSUM: NULL
PARTITION_COMMENT:
    NODEGROUP: NULL
    TABLESPACE_NAME: NULL
2 rows in set (0.00 sec)

```

11.5.17.2.27 PLACEMENT_RULES

The PLACEMENT_RULES table provides information on all explicitly configured Placement Rules in SQL. The information includes both placement policies and directly attached rules.

```
USE information_schema;
DESC placement_rules;
```

Field	Type	Null	Key	Default	Extra
POLICY_ID	bigint(64)	NO		NULL	
CATALOG_NAME	varchar(512)	NO		NULL	
POLICY_NAME	varchar(5)	YES		NULL	
SCHEMA_NAME	varchar(5)	YES		NULL	
TABLE_NAME	varchar(5)	YES		NULL	
PARTITION_NAME	varchar(5)	YES		NULL	
PRIMARY_REGION	varchar(5)	NO		NULL	
REGIONS	varchar(5)	NO		NULL	
CONSTRAINTS	varchar(5)	NO		NULL	
LEADER_CONSTRAINTS	varchar(5)	NO		NULL	
FOLLOWER_CONSTRAINTS	varchar(5)	NO		NULL	
LEARNER_CONSTRAINTS	varchar(5)	NO		NULL	
SCHEDULE	varchar(20)	NO		NULL	
FOLLOWERS	bigint(64)	NO		NULL	
LEARNERS	bigint(64)	NO		NULL	

```
15 rows in set (0.00 sec)
```

Examples

The PLACEMENT_RULES table only shows explicitly configured rules. To see the canonical version of placement rules (including placement policies attached to tables), use the statement SHOW PLACEMENT instead:

```
CREATE TABLE t1 (a INT);
CREATE TABLE t2 (a INT) primary_region="us-east-1" regions="us-east-1";
CREATE PLACEMENT POLICY p1 primary_region="us-east-1" regions="us-east-1";
CREATE TABLE t3 (a INT) PLACEMENT POLICY=p1;
```

```
SHOW PLACEMENT; -- Includes t3.
SELECT * FROM information_schema.placement_rules; -- Does not include t3.
```

```
Query OK, 0 rows affected (0.09 sec)
```

```
Query OK, 0 rows affected (0.11 sec)
```

```
Query OK, 0 rows affected (0.08 sec)
```

```
Query OK, 0 rows affected (0.11 sec)
```

Target	Placement
POLICY p1	PRIMARY_REGION="us-east-1" REGIONS="us-east-1"
TABLE test.t2	PRIMARY_REGION="us-east-1" REGIONS="us-east-1"
TABLE test.t3	PRIMARY_REGION="us-east-1" REGIONS="us-east-1"

3 rows in set (0.00 sec)

POLICY_ID	CATALOG_NAME	POLICY_NAME	SCHEMA_NAME	TABLE_NAME	PARTITION_NAME	PRIMARY_REGION	REGIONS	CONSTRAINTS	LEADER_CONSTRAINTS	FOLLOWER_CONSTRAINTS	LEARNER_CONSTRAINTS	SCHEDULE	FOLLOWERS	LEARNERS	
3	def	p1	NULL	NULL	NULL	us-east-1	us-east-1			0	0				
NULL	def	NULL	test	t2	NULL	us-east-1	us-east-1			0	0				

2 rows in set (0.00 sec)

11.5.17.2.28 PROCESSLIST

PROCESSLIST, just like SHOW PROCESSLIST, is used to view the requests that are being handled.

The PROCESSLIST table has additional columns not present in SHOW PROCESSLIST:

- A MEM column to show the memory used by the request that is being processed, in bytes.
- A TxnStart column to show the start time of the transaction

```
USE information_schema;
DESC processlist;
```

Field	Type	Null	Key	Default	Extra
ID	bigint(21) unsigned	NO		0	
USER	varchar(16)	NO			
HOST	varchar(64)	NO			
DB	varchar(64)	YES		NULL	
COMMAND	varchar(16)	NO			
TIME	int(7)	NO		0	
STATE	varchar(7)	YES		NULL	
INFO	binary(512)	YES		NULL	
MEM	bigint(21) unsigned	YES		NULL	
TxnStart	varchar(64)	NO			

10 rows in set (0.00 sec)

```
SELECT * FROM processlist\G
```

```
***** 1. row *****
ID: 16
USER: root
HOST: 127.0.0.1
DB: information_schema
COMMAND: Query
TIME: 0
STATE: autocommit
INFO: SELECT * FROM processlist
MEM: 0
TxnStart:
1 row in set (0.00 sec)
```

Fields in the PROCESSLIST table are described as follows:

- ID: The ID of the user connection.
- USER: The name of the user who is executing PROCESS.

- HOST: The address that the user is connecting to.
- DB: The name of the currently connected default database.
- COMMAND: The command type that PROCESS is executing.
- TIME: The current execution duration of PROCESS, in seconds.
- STATE: The current connection state.
- INFO: The requested statement that is being processed.
- MEM: The memory used by the request that is being processed, in bytes.
- TxnStart: The start time of the transaction.

CLUSTER_PROCESSLIST

CLUSTER_PROCESSLIST is the cluster system table corresponding to PROCESSLIST. It is used to query the PROCESSLIST information of all TiDB nodes in the cluster. The table schema of CLUSTER_PROCESSLIST has one more column than PROCESSLIST, the INSTANCE column, which stores the address of the TiDB node this row of data is from.

```
SELECT * FROM information_schema.cluster_processlist;
```

INSTANCE	ID	USER	HOST	DB	COMMAND	TIME	STATE	INFO
							MEM	TxnStart
10.0.1.22:10080	150	u1	10.0.1.1	test	Query	0	autocommit	
					select count(*) from usertable	372	05-28	
					03:54:21.230(416976223923077223)			
10.0.1.22:10080	138	root	10.0.1.1	test	Query	0	autocommit	
					SELECT * FROM information_schema.cluster_processlist	0	05-28	
					03:54:21.230(416976223923077220)			
10.0.1.22:10080	151	u1	10.0.1.1	test	Query	0	autocommit	
					select count(*) from usertable	372	05-28	
					03:54:21.230(416976223923077224)			
10.0.1.21:10080	15	u2	10.0.1.1	test	Query	0	autocommit	
					select max(field0) from usertable	496	05-28	
					03:54:21.230(416976223923077222)			
10.0.1.21:10080	14	u2	10.0.1.1	test	Query	0	autocommit	
					select max(field0) from usertable	496	05-28	
					03:54:21.230(416976223923077225)			

11.5.17.2.29 REFERENTIAL_CONSTRAINTS

The REFERENTIAL_CONSTRAINTS table provides information about FOREIGN KEY relationships between tables. Note that TiDB currently does not enforce FOREIGN KEY constraints, or perform actions such as ON DELETE CASCADE.

```
USE information_schema;
DESC referential_constraints;
```

Field	Type	Null	Key	Default	Extra
CONSTRAINT_CATALOG	varchar(512)	NO		NULL	
CONSTRAINT_SCHEMA	varchar(64)	NO		NULL	
CONSTRAINT_NAME	varchar(64)	NO		NULL	
UNIQUE_CONSTRAINT_CATALOG	varchar(512)	NO		NULL	
UNIQUE_CONSTRAINT_SCHEMA	varchar(64)	NO		NULL	
UNIQUE_CONSTRAINT_NAME	varchar(64)	YES		NULL	
MATCH_OPTION	varchar(64)	NO		NULL	
UPDATE_RULE	varchar(64)	NO		NULL	
DELETE_RULE	varchar(64)	NO		NULL	
TABLE_NAME	varchar(64)	NO		NULL	
REFERENCED_TABLE_NAME	varchar(64)	NO		NULL	

11 rows in set (0.00 sec)

```
CREATE TABLE test.parent (
    id INT NOT NULL AUTO_INCREMENT,
    PRIMARY KEY (id)
);

CREATE TABLE test.child (
    id INT NOT NULL AUTO_INCREMENT,
    name varchar(255) NOT NULL,
    parent_id INT DEFAULT NULL,
    PRIMARY KEY (id),
    CONSTRAINT fk_parent FOREIGN KEY (parent_id) REFERENCES parent (id) ON
        UPDATE CASCADE ON DELETE RESTRICT
);
```

```
SELECT * FROM referential_constraints\G
```

```
***** 1. row *****
CONSTRAINT_CATALOG: def
CONSTRAINT_SCHEMA: test
CONSTRAINT_NAME: fk_parent
UNIQUE_CONSTRAINT_CATALOG: def
UNIQUE_CONSTRAINT_SCHEMA: test
UNIQUE_CONSTRAINT_NAME: PRIMARY
MATCH_OPTION: NONE
UPDATE_RULE: CASCADE
DELETE_RULE: RESTRICT
TABLE_NAME: child
REFERENCED_TABLE_NAME: parent
1 row in set (0.00 sec)
```

11.5.17.2.30 SCHEMATA

The SCHEMATA table provides information about databases. The table data is equivalent to the result of the SHOW DATABASES statement.

```
USE information_schema;
desc SCHEMATA;
```

Field	Type	Null	Key	Default	Extra
CATALOG_NAME	varchar(512)	YES		NULL	
SCHEMA_NAME	varchar(64)	YES		NULL	
DEFAULT_CHARACTER_SET_NAME	varchar(64)	YES		NULL	
DEFAULT_COLLATION_NAME	varchar(32)	YES		NULL	
SQL_PATH	varchar(512)	YES		NULL	

```
5 rows in set (0.00 sec)
```

```
SELECT * FROM SCHEMATA;
```

CATALOG_NAME	SCHEMA_NAME	DEFAULT_CHARACTER_SET_NAME	DEFAULT_COLLATION_NAME	SQL_PATH
--------------	-------------	----------------------------	------------------------	----------

→			
def	INFORMATION_SCHEMA	utf8mb4	utf8mb4_bin
→	NULL		
def	METRICS_SCHEMA	utf8mb4	utf8mb4_bin
→	NULL		
def	mysql	utf8mb4	utf8mb4_bin
→	NULL		
def	PERFORMANCE_SCHEMA	utf8mb4	utf8mb4_bin
→	NULL		
def	test	utf8mb4	utf8mb4_bin
→	NULL		
→			
5 rows in set (0.00 sec)			

Fields in the SCHEMATA table are described as follows:

- CATALOG_NAME: The catalog to which the database belongs.
- SCHEMA_NAME: The database name.
- DEFAULT_CHARACTER_SET_NAME: The default character set of the database.
- DEFAULT_COLLATION_NAME: The default collation of the database.
- SQL_PATH: The value of this item is always NULL.

11.5.17.2.31 SEQUENCES

The SEQUENCES table provides information about sequences. The [sequences feature](#) is modeled on a similar feature in MariaDB.

```
USE information_schema;
DESC sequences;
```

Field	Type	Null	Key	Default	Extra
TABLE_CATALOG	varchar(512)	NO		NULL	
SEQUENCE_SCHEMA	varchar(64)	NO		NULL	
SEQUENCE_NAME	varchar(64)	NO		NULL	
CACHE	tinyint(4)	NO		NULL	
CACHE_VALUE	bigint(21)	YES		NULL	
CYCLE	tinyint(4)	NO		NULL	
INCREMENT	bigint(21)	NO		NULL	
MAX_VALUE	bigint(21)	YES		NULL	
MIN_VALUE	bigint(21)	YES		NULL	
START	bigint(21)	YES		NULL	

```
+-----+-----+-----+-----+-----+
| COMMENT | varchar(64) | YES | NULL |       |
+-----+-----+-----+-----+-----+
11 rows in set (0.00 sec)
```

```
CREATE SEQUENCE test.seq;
SELECT nextval(test.seq);
SELECT * FROM sequences\G
```

```
+-----+
| nextval(test.seq) |
+-----+
|          1 |
+-----+
1 row in set (0.01 sec)
```

```
***** 1. row *****
TABLE_CATALOG: def
SEQUENCE_SCHEMA: test
SEQUENCE_NAME: seq
CACHE: 1
CACHE_VALUE: 1000
CYCLE: 0
INCREMENT: 1
MAX_VALUE: 9223372036854775806
MIN_VALUE: 1
START: 1
COMMENT:
1 row in set (0.00 sec)
```

11.5.17.2.32 SESSION_VARIABLES

The SESSION_VARIABLES table provides information about session variables. The table data is similar to the result of the SHOW SESSION VARIABLES statement.

```
USE information_schema;
DESC session_variables;
```

```
+-----+-----+-----+-----+-----+-----+
| Field      | Type       | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| VARIABLE_NAME | varchar(64) | YES |   | NULL |   |
| VARIABLE_VALUE | varchar(1024) | YES |   | NULL |   |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

```
SELECT * FROM session_variables ORDER BY variable_name LIMIT 10;
```

VARIABLE_NAME	VARIABLE_VALUE
allow_auto_random_explicit_insert	off
auto_increment_increment	1
auto_increment_offset	1
autocommit	1
automatic_sp_privileges	1
avoid_temporal_upgrade	0
back_log	80
basedir	/usr/local/mysql
big_tables	0
bind_address	*

10 rows in set (0.00 sec)

The description of columns in the SESSION_VARIABLES table is as follows:

- VARIABLE_NAME: The name of the session-level variable in the database.
- VARIABLE_VALUE: The value of the session-level variable in the database.

11.5.17.2.33 SLOW_QUERY

The SLOW_QUERY table provides the slow query information of the current node, which is the parsing result of the TiDB slow log file. The column names in the table are corresponding to the field names in the slow log. For how to use this table to identify problematic statements and improve query performance, see [Slow Query Log Document](#).

```
USE information_schema;
DESC slow_query;
```

Field	Type	Null	Key	Default	Extra
Time	timestamp(6)	YES		NULL	
Txn_start_ts	bigint(20) unsigned	YES		NULL	
User	varchar(64)	YES		NULL	
Host	varchar(64)	YES		NULL	
Conn_ID	bigint(20) unsigned	YES		NULL	
Query_time	double	YES		NULL	

Parse_time	double	YES	NULL		
Compile_time	double	YES	NULL		
Rewrite_time	double	YES	NULL		
Preproc_subqueries	bigint(20) unsigned	YES	NULL		
Preproc_subqueries_time	double	YES	NULL		
Optimize_time	double	YES	NULL		
Wait_TS	double	YES	NULL		
Prewrite_time	double	YES	NULL		
Wait_prewrite_binlog_time	double	YES	NULL		
Commit_time	double	YES	NULL		
Get_commit_ts_time	double	YES	NULL		
Commit_backoff_time	double	YES	NULL		
Backoff_types	varchar(64)	YES	NULL		
Resolve_lock_time	double	YES	NULL		
Local_latch_wait_time	double	YES	NULL		
Write_keys	bigint(22)	YES	NULL		
Write_size	bigint(22)	YES	NULL		
Prewrite_region	bigint(22)	YES	NULL		
Txn_retry	bigint(22)	YES	NULL		
Cop_time	double	YES	NULL		
Process_time	double	YES	NULL		
Wait_time	double	YES	NULL		
Backoff_time	double	YES	NULL		
LockKeys_time	double	YES	NULL		
Request_count	bigint(20) unsigned	YES	NULL		
Total_keys	bigint(20) unsigned	YES	NULL		
Process_keys	bigint(20) unsigned	YES	NULL		
DB	varchar(64)	YES	NULL		
Index_names	varchar(100)	YES	NULL		
Is_internal	tinyint(1)	YES	NULL		
Digest	varchar(64)	YES	NULL		
Stats	varchar(512)	YES	NULL		
Cop_proc_avg	double	YES	NULL		
Cop_proc_p90	double	YES	NULL		
Cop_proc_max	double	YES	NULL		
Cop_proc_addr	varchar(64)	YES	NULL		
Cop_wait_avg	double	YES	NULL		
Cop_wait_p90	double	YES	NULL		
Cop_wait_max	double	YES	NULL		
Cop_wait_addr	varchar(64)	YES	NULL		
Mem_max	bigint(20)	YES	NULL		
Disk_max	bigint(20)	YES	NULL		
Succ	tinyint(1)	YES	NULL		
Plan_from_cache	tinyint(1)	YES	NULL		
Plan	longblob	YES	NULL		

Plan_digest	varchar(128)	YES	NULL		
Prev_stmt	longblob	YES	NULL		
Query	longblob	YES	NULL		
+-----+-----+-----+-----+-----+					
→					
54 rows in set (0.00 sec)					

CLUSTER_SLOW_QUERY table

The CLUSTER_SLOW_QUERY table provides the slow query information of all nodes in the cluster, which is the parsing result of the TiDB slow log files. You can use the CLUSTER_SLOW_QUERY table the way you do with SLOW_QUERY. The table schema of the CLUSTER_SLOW_QUERY table differs from that of the SLOW_QUERY table in that an INSTANCE column is added to CLUSTER_SLOW_QUERY. The INSTANCE column represents the TiDB node address of the row information on the slow query. For how to use this table to identify problematic statements and improve query performance, see [Slow Query Log Document](#).

```
desc cluster_slow_query;
```

+--	Field	Type	Null	Key	Default	Extra	
+--	INSTANCE	varchar(64)	YES		NULL		
→	Time	timestamp(6)	YES		NULL		
→	Txn_start_ts	bigint(20) unsigned	YES		NULL		
→	User	varchar(64)	YES		NULL		
→	Host	varchar(64)	YES		NULL		
→	Conn_ID	bigint(20) unsigned	YES		NULL		
→	Query_time	double	YES		NULL		
→	Parse_time	double	YES		NULL		
→	Compile_time	double	YES		NULL		
→	Rewrite_time	double	YES		NULL		
→	Preproc_subqueries	bigint(20) unsigned	YES		NULL		
→	Preproc_subqueries_time	double	YES		NULL		
→	Optimize_time	double	YES		NULL		
→	Wait_TS	double	YES		NULL		
→	Prewrite_time	double	YES		NULL		
→	Wait_prewrite_binlog_time	double	YES		NULL		
→	Commit_time	double	YES		NULL		
→	Get_commit_ts_time	double	YES		NULL		
→	Commit_backoff_time	double	YES		NULL		
→	Backoff_types	varchar(64)	YES		NULL		

Resolve_lock_time	double	YES	NULL		
Local_latch_wait_time	double	YES	NULL		
Write_keys	bigint(22)	YES	NULL		
Write_size	bigint(22)	YES	NULL		
Prewrite_region	bigint(22)	YES	NULL		
Txn_retry	bigint(22)	YES	NULL		
Cop_time	double	YES	NULL		
Process_time	double	YES	NULL		
Wait_time	double	YES	NULL		
Backoff_time	double	YES	NULL		
LockKeys_time	double	YES	NULL		
Request_count	bigint(20) unsigned	YES	NULL		
Total_keys	bigint(20) unsigned	YES	NULL		
Process_keys	bigint(20) unsigned	YES	NULL		
DB	varchar(64)	YES	NULL		
Index_names	varchar(100)	YES	NULL		
Is_internal	tinyint(1)	YES	NULL		
Digest	varchar(64)	YES	NULL		
Stats	varchar(512)	YES	NULL		
Cop_proc_avg	double	YES	NULL		
Cop_proc_p90	double	YES	NULL		
Cop_proc_max	double	YES	NULL		
Cop_proc_addr	varchar(64)	YES	NULL		
Cop_wait_avg	double	YES	NULL		
Cop_wait_p90	double	YES	NULL		
Cop_wait_max	double	YES	NULL		
Cop_wait_addr	varchar(64)	YES	NULL		
Mem_max	bigint(20)	YES	NULL		
Disk_max	bigint(20)	YES	NULL		
Succ	tinyint(1)	YES	NULL		
Plan_from_cache	tinyint(1)	YES	NULL		
Plan	longblob	YES	NULL		
Plan_digest	varchar(128)	YES	NULL		
Prev_stmt	longblob	YES	NULL		
Query	longblob	YES	NULL		
+--					
→	-----+-----+-----+-----+-----+				
→					
55 rows in set (0.00 sec)					

When the cluster system table is queried, TiDB does not obtain data from all nodes, but pushes down the related calculation to other nodes. The execution plan is as follows:

```
desc SELECT count(*) FROM cluster_slow_query WHERE user = 'u1';
```

id	estRows	task	access object	
operator info				
StreamAgg_20	1.00	root		funcs:
↳ count(Column#53)->Column#51				
TableReader_21	1.00	root		data:
↳ StreamAgg_9				
↳ StreamAgg_9	1.00	cop[tidb]		funcs
↳ :count(1)->Column#53				
↳ Selection_19	10.00	cop[tidb]		eq()
↳ information_schema.cluster_slow_query.user, "u1")				
↳ TableFullScan_18	10000.00	cop[tidb]	table:CLUSTER_SLOW_QUERY	
↳ keep order:false, stats:pseudo				

In the above execution plan, the `user = u1` condition is pushed down to other (`cop`) TiDB nodes, and the aggregate operator is also pushed down (the `StreamAgg` operator in the graph).

Currently, because statistics of the system tables are not collected, sometimes some aggregation operators cannot be pushed down, which results in slow execution. In this case, you can manually specify the SQL HINT to push down the aggregation operators. For example:

```
SELECT /*+ AGG_TO_COP() */ count(*) FROM cluster_slow_query GROUP BY user;
```

11.5.17.2.34 STATISTICS

The `STATISTICS` table provides information about table indexes.

```
USE information_schema;
DESC statistics;
```

Field	Type	Null	Key	Default	Extra
TABLE_CATALOG	varchar(512)	YES		NULL	
TABLE_SCHEMA	varchar(64)	YES		NULL	
TABLE_NAME	varchar(64)	YES		NULL	
NON_UNIQUE	varchar(1)	YES		NULL	
INDEX_SCHEMA	varchar(64)	YES		NULL	
INDEX_NAME	varchar(64)	YES		NULL	

SEQ_IN_INDEX	bigint(2)	YES	NULL		
COLUMN_NAME	varchar(21)	YES	NULL		
COLLATION	varchar(1)	YES	NULL		
CARDINALITY	bigint(21)	YES	NULL		
SUB_PART	bigint(3)	YES	NULL		
PACKED	varchar(10)	YES	NULL		
NULLABLE	varchar(3)	YES	NULL		
INDEX_TYPE	varchar(16)	YES	NULL		
COMMENT	varchar(16)	YES	NULL		
INDEX_COMMENT	varchar(1024)	YES	NULL		
IS_VISIBLE	varchar(3)	YES	NULL		
Expression	varchar(64)	YES	NULL		

18 rows in set (0.00 sec)

Fields in the STATISTICS table are described as follows:

- TABLE_CATALOG: The name of the catalog to which the table containing the index belongs. This value is always `def`.
- TABLE_SCHEMA: The name of the database to which the table containing the index belongs.
- TABLE_NAME: The name of the table containing the index.
- NON_UNIQUE: If the index must not contain duplicate values, the value is 0; if duplicate values are allowed in the index, the value is 1.
- INDEX_SCHEMA: The name of the database to which the index belongs.
- INDEX_NAME: The name of the index. If the index is the primary key, then the value is always PRIMARY.
- SEQ_IN_INDEX: The column number in the index, starting from 1.
- COLUMN_NAME: The column name. See the description of the Expression column.
- COLLATION: The sorting method of the columns in the index. The value can be A (ascending order), D (descending order) or NULL (unsorted).
- CARDINALITY: TiDB does not use this field. The field value is always 0.
- SUB_PART: The prefix of the index. If only part of the prefix of the column is indexed, the value is the number of indexed characters; if the entire column is indexed, the value is NULL.
- PACKED: TiDB does not use this field. This value is always NULL.
- NULLABLE: If the column might contain a NULL value, the value is YES; if not, the value is ''.
- INDEX_TYPE: The type of the index.
- COMMENT: Other information related to the index.
- INDEX_COMMENT: Any comment with comment attribute provided for the index when creating the index.
- IS_VISIBLE: Whether the optimizer can use this index.
- Expression For the index key of the non-expression part, this value is NULL; for the index key of the expression part, this value is the expression itself. Refer to [Expression](#)

Index.

The following statements are equivalent:

```
SELECT * FROM INFORMATION_SCHEMA.STATISTICS
  WHERE table_name = 'tbl_name'
    AND table_schema = 'db_name'

SHOW INDEX
  FROM tbl_name
  FROM db_name
```

11.5.17.2.35 TABLES

The TABLES table provides information about tables in databases:

```
USE information_schema;
DESC tables;
```

Field	Type	Null	Key	Default	Extra
TABLE_CATALOG	varchar(512)	YES		NULL	
TABLE_SCHEMA	varchar(64)	YES		NULL	
TABLE_NAME	varchar(64)	YES		NULL	
TABLE_TYPE	varchar(64)	YES		NULL	
ENGINE	varchar(64)	YES		NULL	
VERSION	bigint(21)	YES		NULL	
ROW_FORMAT	varchar(10)	YES		NULL	
TABLE_ROWS	bigint(21)	YES		NULL	
AVG_ROW_LENGTH	bigint(21)	YES		NULL	
DATA_LENGTH	bigint(21)	YES		NULL	
MAX_DATA_LENGTH	bigint(21)	YES		NULL	
INDEX_LENGTH	bigint(21)	YES		NULL	
DATA_FREE	bigint(21)	YES		NULL	
AUTO_INCREMENT	bigint(21)	YES		NULL	
CREATE_TIME	datetime	YES		NULL	
UPDATE_TIME	datetime	YES		NULL	
CHECK_TIME	datetime	YES		NULL	
TABLE_COLLATION	varchar(32)	NO		utf8_bin	
CHECKSUM	bigint(21)	YES		NULL	

```

| CREATE_OPTIONS          | varchar(255) | YES | NULL |
| TABLE_COMMENT           | varchar(2048) | YES | NULL |
| TIDB_TABLE_ID           | bigint(21)   | YES | NULL |
| TIDB_ROW_ID_SHARDING_INFO | varchar(255) | YES | NULL |
+---+
→ -----+-----+-----+-----+
→
23 rows in set (0.00 sec)

```

```
SELECT * FROM tables WHERE table_schema='mysql' AND table_name='user' \G
```

```
***** 1. row *****
TABLE_CATALOG: def
TABLE_SCHEMA: mysql
TABLE_NAME: user
TABLE_TYPE: BASE TABLE
ENGINE: InnoDB
VERSION: 10
ROW_FORMAT: Compact
TABLE_ROWS: 0
AVG_ROW_LENGTH: 0
DATA_LENGTH: 0
MAX_DATA_LENGTH: 0
INDEX_LENGTH: 0
DATA_FREE: 0
AUTO_INCREMENT: NULL
CREATE_TIME: 2020-07-05 09:25:51
UPDATE_TIME: NULL
CHECK_TIME: NULL
TABLE_COLLATION: utf8mb4_bin
CHECKSUM: NULL
CREATE_OPTIONS:
TABLE_COMMENT:
TIDB_TABLE_ID: 5
TIDB_ROW_ID_SHARDING_INFO: NULL
1 row in set (0.00 sec)
```

The following statements are equivalent:

```

SELECT table_name FROM INFORMATION_SCHEMA.TABLES
WHERE table_schema = 'db_name'
[AND table_name LIKE 'wild']

SHOW TABLES
FROM db_name

```

```
[LIKE 'wild']
```

The description of columns in the TABLES table is as follows:

- TABLE_CATALOG: The name of the catalog which the table belongs to. The value is always def.
- TABLE_SCHEMA: The name of the schema which the table belongs to.
- TABLE_NAME: The name of the table.
- TABLE_TYPE: The type of the table.
- ENGINE: The type of the storage engine. The value is currently InnoDB.
- VERSION: Version. The value is 10 by default.
- ROW_FORMAT: The row format. The value is currently Compact.
- TABLE_ROWS: The number of rows in the table in statistics.
- AVG_ROW_LENGTH: The average row length of the table. $AVG_ROW_LENGTH = DATA_LENGTH / TABLE_ROWS$.
- DATA_LENGTH: Data length. $DATA_LENGTH = TABLE_ROWS * \text{the sum of storage lengths of the columns in the tuple}$. The replicas of TiKV are not taken into account.
- MAX_DATA_LENGTH: The maximum data length. The value is currently 0, which means the data length has no upper limit.
- INDEX_LENGTH: The index length. $INDEX_LENGTH = TABLE_ROWS * \text{the sum of lengths of the columns in the index tuple}$. The replicas of TiKV are not taken into account.
- DATA_FREE: Data fragment. The value is currently 0.
- AUTO_INCREMENT: The current step of the auto-increment primary key.
- CREATE_TIME: The time at which the table is created.
- UPDATE_TIME: The time at which the table is updated.
- CHECK_TIME: The time at which the table is checked.
- TABLE_COLLATION: The collation of strings in the table.
- CHECKSUM: Checksum.
- CREATE_OPTIONS: Creates options.
- TABLE_COMMENT: The comments and notes of the table.

Most of the information in the table is the same as MySQL. Only two columns are newly defined by TiDB:

- TIDB_TABLE_ID: to indicate the internal ID of a table. This ID is unique in a TiDB cluster.
- TIDDB_ROW_ID_SHARDING_INFO: to indicate the sharding type of a table. The possible values are as follows:
 - "NOT_SHARDED": the table is not sharded.
 - "NOT_SHARDED(PK_IS_HANDLE)": the table that defines an integer Primary Key as its row id is not sharded.
 - "PK_AUTO_RANDOM_BITS={bit_number)": the table that defines an integer Primary Key as its row id is sharded because the Primary Key is assigned with AUTO_RANDOM attribute.

- "SHARD_BITS={bit_number)": the table is sharded using SHARD_ROW_ID_BITS={ \rightarrow bit_number}.
- NULL: the table is a system table or view, and thus cannot be sharded.

11.5.17.2.36 TABLE_CONSTRAINTS

The TABLE_CONSTRAINTS table describes which tables have constraints.

```
USE information_schema;
DESC table_constraints;
```

Field	Type	Null	Key	Default	Extra
CONSTRAINT_CATALOG	varchar(512)	YES		NULL	
CONSTRAINT_SCHEMA	varchar(64)	YES		NULL	
CONSTRAINT_NAME	varchar(64)	YES		NULL	
TABLE_SCHEMA	varchar(64)	YES		NULL	
TABLE_NAME	varchar(64)	YES		NULL	
CONSTRAINT_TYPE	varchar(64)	YES		NULL	

6 rows in set (0.00 sec)

```
SELECT * FROM table_constraints WHERE constraint_type='UNIQUE';
```

CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_TYPE
def	mysql	name	mysql		UNIQUE
def	mysql	help_topic	mysql		UNIQUE
def	mysql	tbl	mysql		UNIQUE
def	mysql	stats_meta	mysql		UNIQUE
def	mysql	tbl	mysql		UNIQUE
def	mysql	stats_histograms	mysql		UNIQUE
def	mysql	tbl	mysql		UNIQUE
def	mysql	stats_buckets	mysql		UNIQUE
def	mysql	delete_range_index	mysql		UNIQUE
def	mysql	gc_delete_range	mysql		UNIQUE
def	mysql	delete_range_done_index	mysql		UNIQUE
def	mysql	gc_delete_range_done	mysql		UNIQUE

```
| def | PERFORMANCE_SCHEMA | SCHEMA_NAME |  
    ↳ PERFORMANCE_SCHEMA | events_statements_summary_by_digest | UNIQUE |  
+--  
    ↳ -----+-----+-----+  
    ↳  
7 rows in set (0.01 sec)
```

Fields in the TABLE_CONSTRAINTS table are described as follows:

- CONSTRAINT_CATALOG: The name of the catalog to which the constraint belongs. This value is always `def`.
- CONSTRAINT_SCHEMA: The name of the database to which the constraint belongs.
- CONSTRAINT_NAME: The name of the constraint.
- TABLE_NAME: The name of the table.
- CONSTRAINT_TYPE: The type of the constraint. The value can be `UNIQUE`, `PRIMARY KEY` or `FOREIGN KEY`. The `UNIQUE` and `PRIMARY KEY` information is similar to the execution result of the `SHOW INDEX` statement.

11.5.17.2.37 TABLE_STORAGE_STATS

The TABLE_STORAGE_STATS table provides information about table sizes as stored by the storage engine (TiKV).

```
USE information_schema;  
DESC table_storage_stats;
```

Field	Type	Null	Key	Default	Extra
TABLE_SCHEMA	varchar(64)	YES		NULL	
TABLE_NAME	varchar(64)	YES		NULL	
TABLE_ID	bigint(21)	YES		NULL	
PEER_COUNT	bigint(21)	YES		NULL	
REGION_COUNT	bigint(21)	YES		NULL	
EMPTY_REGION_COUNT	bigint(21)	YES		NULL	
TABLE_SIZE	bigint(64)	YES		NULL	
TABLE_KEYS	bigint(64)	YES		NULL	

```
8 rows in set (0.00 sec)
```

```
CREATE TABLE test.t1 (id INT);  
INSERT INTO test.t1 VALUES (1);  
SELECT * FROM table_storage_stats WHERE table_schema = 'test' AND  
    ↳ table_name = 't1'\G
```

```
***** 1. row *****
  TABLE_SCHEMA: test
    TABLE_NAME: t1
      TABLE_ID: 56
    PEER_COUNT: 1
    REGION_COUNT: 1
EMPTY_REGION_COUNT: 1
    TABLE_SIZE: 1
    TABLE_KEYS: 0
1 row in set (0.00 sec)
```

11.5.17.2.38 TIDB_HOT_REGIONS

The TIDB_HOT_REGIONS table provides information about hotspot Regions.

```
USE information_schema;
DESC tidb_hot_regions;
```

Field	Type	Null	Key	Default	Extra
TABLE_ID	bigint(21)	YES		NULL	
INDEX_ID	bigint(21)	YES		NULL	
DB_NAME	varchar(64)	YES		NULL	
TABLE_NAME	varchar(64)	YES		NULL	
INDEX_NAME	varchar(64)	YES		NULL	
REGION_ID	bigint(21)	YES		NULL	
TYPE	varchar(64)	YES		NULL	
MAX_HOT_DEGREE	bigint(21)	YES		NULL	
REGION_COUNT	bigint(21)	YES		NULL	
FLOW_BYTES	bigint(21)	YES		NULL	

10 rows in set (0.00 sec)

The description of columns in the TIDB_HOT_REGIONS table is as follows:

- TABLE_ID: ID of located.
- INDEX_ID: ID of the table in which the hot Region is the index in which the hot Region is located.
- DB_NAME: The database name of the object in which the hot Region is located.
- TABLE_NAME: The name of the table in which the hot Region is located.
- INDEX_NAME: The name of the index in which the hot Region is located.
- REGION_ID: ID of the hot Region.
- TYPE: The type of the hot Region.

- `MAX_HOT_DEGREE`: The maximum hot degree of the Region.
- `REGION_COUNT`: The number of Regions in the instance.
- `FLOW_BYTES`: The number of bytes written and read in the Region.

11.5.17.2.39 TIDB_INDEXES

The `TIDB_INDEXES` table provides the INDEX information of all tables.

```
USE information_schema;
DESC tidb_indexes;
```

Field	Type	Null	Key	Default	Extra
<code>TABLE_SCHEMA</code>	<code>varchar(64)</code>	YES		<code>NULL</code>	
<code>TABLE_NAME</code>	<code>varchar(64)</code>	YES		<code>NULL</code>	
<code>NON_UNIQUE</code>	<code>bigint(21)</code>	YES		<code>NULL</code>	
<code>KEY_NAME</code>	<code>varchar(64)</code>	YES		<code>NULL</code>	
<code>SEQ_IN_INDEX</code>	<code>bigint(21)</code>	YES		<code>NULL</code>	
<code>COLUMN_NAME</code>	<code>varchar(64)</code>	YES		<code>NULL</code>	
<code>SUB_PART</code>	<code>bigint(21)</code>	YES		<code>NULL</code>	
<code>INDEX_COMMENT</code>	<code>varchar(2048)</code>	YES		<code>NULL</code>	
<code>Expression</code>	<code>varchar(64)</code>	YES		<code>NULL</code>	
<code>INDEX_ID</code>	<code>bigint(21)</code>	YES		<code>NULL</code>	

10 rows in set (0.00 sec)

`INDEX_ID` is the unique ID that TiDB allocates for each index. It can be used to do a join operation with `INDEX_ID` obtained from another table or API.

For example, you can obtain `TABLE_ID` and `INDEX_ID` that are involved in some slow query in the `SLOW_QUERY table` and then obtain the specific index information using the following SQL statements:

```
SELECT
tidb_indexes.*
FROM
tidb_indexes,
tables
WHERE
tidb_indexes.table_schema = tables.table_schema
AND tidb_indexes.table_name = tidb_indexes.table_name
AND tables.tidb_table_id = ?
AND index_id = ?
```

Fields in the `TIDB_INDEXES` table are described as follows:

- TABLE_SCHEMA: The name of the schema to which the index belongs.
- TABLE_NAME: The name of the table to which the index belongs.
- NON_UNIQUE: If the index is unique, the value is 0; otherwise, the value is 1.
- KEY_NAME: The index name. If the index is the primary key, the name is PRIMARY.
- SEQ_IN_INDEX: The sequential number of columns in the index, which starts from 1.
- COLUMN_NAME: The name of the column where the index is located.
- SUB_PART: The prefix length of the index. If the column is partly indexed, the SUB_PART value is the count of the indexed characters; otherwise, the value is NULL.
- INDEX_COMMENT: The comment of the index, which is made when the index is created.
- INDEX_ID: The index ID.

11.5.17.2.40 TIDB_SERVERS_INFO

The TIDB_SERVERS_INFO table provides information about TiDB servers in the TiDB Cluster (namely, tidb-server processes).

```
USE information_schema;
DESC tidb_servers_info;
```

Field	Type	Null	Key	Default	Extra
DDL_ID	varchar(64)	YES		NULL	
IP	varchar(64)	YES		NULL	
PORT	bigint(21)	YES		NULL	
STATUS_PORT	bigint(21)	YES		NULL	
LEASE	varchar(64)	YES		NULL	
VERSION	varchar(64)	YES		NULL	
GIT_HASH	varchar(64)	YES		NULL	
BINLOG_STATUS	varchar(64)	YES		NULL	

8 rows in set (0.00 sec)

```
SELECT * FROM tidb_servers_info\G
```

```
***** 1. row *****
DDL_ID: 771c169d-0a3a-48ea-a93c-a4d6751d3674
IP: 0.0.0.0
PORT: 4000
STATUS_PORT: 10080
LEASE: 45s
VERSION: 5.7.25-TiDB-v4.0.0-beta.2-720-g0df3b74f5
GIT_HASH: 0df3b74f55f8f8fbde39bbd5d471783f49dc10f7
BINLOG_STATUS: Off
1 row in set (0.00 sec)
```

11.5.17.2.41 TIDB_TRX

The TIDB_TRX table provides information about the transactions currently being executed on the TiDB node.

```
USE information_schema;
DESC tidb_trx;
```

Field	Type	Null	Key
Default	Extra		
ID	bigint(21) unsigned	NO	PRI NULL
START_TIME	timestamp(6)	YES	NULL
CURRENT_SQL_DIGEST	varchar(64)	YES	NULL
CURRENT_SQL_DIGEST_TEXT	text	YES	NULL
STATE	enum('Idle','Running','LockWaiting','Committing','RollingBack')	YES	NULL
WAITING_START_TIME	timestamp(6)	YES	NULL
MEM_BUFFER_KEYS	bigint(64)	YES	NULL
MEM_BUFFER_BYTES	bigint(64)	YES	NULL
SESSION_ID	bigint(21) unsigned	YES	NULL
USER	varchar(16)	YES	NULL
DB	varchar(64)	YES	NULL

→					
ALL_SQL_DIGESTS		text			
→				YES	NULL
→					
+--					
→ -----+-----+					
→					

The meaning of each column field in the `TIDB_TRX` table is as follows:

- **ID**: The transaction ID, which is the `start_ts` (start timestamp) of the transaction.
- **START_TIME**: The start time of the transaction, which is the physical time corresponding to the `start_ts` of the transaction.
- **CURRENT_SQL_DIGEST**: The digest of the SQL statement currently being executed in the transaction.
- **CURRENT_SQL_DIGEST_TEXT**: The normalized form of the SQL statement currently being executed by the transaction, that is, the SQL statement without arguments and format. It corresponds to `CURRENT_SQL_DIGEST`.
- **STATE**: The current state of the transaction. The possible values include:
 - **Idle**: The transaction is in an idle state, that is, it is waiting for the user to input a query.
 - **Running**: The transaction is executing a query.
 - **LockWaiting**: The transaction is waiting for the pessimistic lock to be acquired. Note that the transaction enters this state at the beginning of the pessimistic locking operation, no matter whether it is blocked by other transactions or not.
 - **Committing**: The transaction is in the process of commit.
 - **RollingBack**: The transaction is being rolled back.
- **WAITING_START_TIME**: When the value of `STATE` is `LockWaiting`, this column shows the start time of the waiting.
- **MEM_BUFFER_KEYS**: The number of keys written into the memory buffer by the current transaction.
- **MEM_BUFFER_BYTES**: The total number of key-value bytes written into the memory buffer by the current transaction.
- **SESSION_ID**: The ID of the session to which this transaction belongs.
- **USER**: The name of the user who performs the transaction.
- **DB**: The current default database name of the session in which the transaction is executed.
- **ALL_SQL_DIGESTS**: The digest list of statements that have been executed by the transaction. The list is shown as a string array in JSON format. Each transaction records at most the first 50 statements. Using the `TIDB_DECODE_SQL_DIGESTS` function, you can convert the information in this column into a list of corresponding normalized SQL statements.

Note:

- Only users with the `PROCESS` privilege can obtain the complete information in this table. Users without the `PROCESS` privilege can only query information of the transactions performed by the current user.
- The information (SQL digest) in the `CURRENT_SQL_DIGEST` and `ALL_SQL_DIGESTS` columns is the hash value calculated from the normalized SQL statement. The information in the `CURRENT_SQL_DIGEST_TEXT` column and the result returned from the `TIDB_DECODE_SQL_DIGESTS` function are internally queried from the statements summary tables, so it is possible that the corresponding statement cannot be found internally. For the detailed description of SQL digests and the statements summary tables, see [Statement Summary Tables](#).
- The `TIDB_DECODE_SQL_DIGESTS` function call has a high overhead. If the function is called to query historical SQL statements for a large number of transactions, the query might take a long time. If the cluster is large with many concurrent transactions, avoid directly using this function on the `ALL_SQL_DIGEST` column while querying the full table of `TIDB_TRX`. This means to avoid an SQL statement like `select *, tidb_decode_sql_digests(all_sql_digests) from tidb_trx`.
- Currently the `TIDB_TRX` table does not support showing information of TiDB internal transactions.

Example

```
select * from information_schema.tidb_trx\G
```

```
***** 1. row *****
    ID: 426789913200689153
    START_TIME: 2021-08-04 10:51:54.883000
    CURRENT_SQL_DIGEST: NULL
    CURRENT_SQL_DIGEST_TEXT: NULL
        STATE: Idle
    WAITING_START_TIME: NULL
        MEM_BUFFER_KEYS: 1
        MEM_BUFFER_BYTES: 29
            SESSION_ID: 7
                USER: root
                DB: test
    ALL_SQL_DIGESTS: [
        ↪ e6f07d43b5c21db0fb9a31feac2dc599787763393dd5acbfad80e247eb02ad5
        ↪ ", "04
```

```

    ↳ fa858fa491c62d194faec2ab427261cc7998b3f1ccf8f6844febca504cb5e9
    ↳ ","
    ↳ b83710fa8ab7df8504920e8569e48654f621cf828afbe7527fd003b79f48da9e
    ↳ "]"
***** 2. row *****
    ID: 426789921471332353
    START_TIME: 2021-08-04 10:52:26.433000
    CURRENT_SQL_DIGEST: 38
    ↳ b03afa5debbdf0326a014dbe5012a62c51957f1982b3093e748460f8b00821
CURRENT_SQL_DIGEST_TEXT: update `t` set `v` = `v` + ? where `id` = ?
    STATE: LockWaiting
    WAITING_START_TIME: 2021-08-04 10:52:35.106568
    MEM_BUFFER_KEYS: 0
    MEM_BUFFER_BYTES: 0
    SESSION_ID: 9
    USER: root
    DB: test
    ALL_SQL_DIGESTS: [
        ↳ e6f07d43b5c21db0fbb9a31feac2dc599787763393dd5acbfad80e247eb02ad5
        ↳ ", "38
        ↳ b03afa5debbdf0326a014dbe5012a62c51957f1982b3093e748460f8b00821
        ↳ "]"
2 rows in set (0.01 sec)

```

From the query result of this example, you can see that: the current node has two on-going transactions. One transaction is in the idle state (STATE is Idle and CURRENT_SQL_DIGEST is NULL), and this transaction has executed 3 statements (there are three records in the ALL_SQL_DIGESTS list, which are the digests of the three SQL statements that have been executed). Another transaction is executing a statement and waiting for the lock (STATE is LockWaiting and WAITING_START_TIME shows the start time of the waiting lock). The transaction has executed 2 statements, and the statement currently being executed is in the form of "update `t` set `v` = `v` + ? where `id` = ?".

```

select id, all_sql_digests, tidb_decode_sql_digests(all_sql_digests) as
    ↳ all_sqls from information_schema.tidb_trx\G

```

```

***** 1. row *****
    id: 426789913200689153
all_sql_digests: [
    ↳ e6f07d43b5c21db0fbb9a31feac2dc599787763393dd5acbfad80e247eb02ad5", "04
    ↳ fa858fa491c62d194faec2ab427261cc7998b3f1ccf8f6844febca504cb5e9",
    ↳ b83710fa8ab7df8504920e8569e48654f621cf828afbe7527fd003b79f48da9e"]
    all_sqls: ["begin", "insert into `t` values ( ... )", "update `t` set `v` = `v` + ?"]
***** 2. row *****

```

```

    id: 426789921471332353
all_sql_digests: [
    ↪ e6f07d43b5c21db0fbb9a31feac2dc599787763393dd5acbfad80e247eb02ad5", "38
    ↪ b03afa5debbdf0326a014dbe5012a62c51957f1982b3093e748460f8b00821"]
    all_sqls: ["begin", "update `t` set `v` = `v` + ? where `id` = ?"]

```

This query calls the `TIDB_DECODE_SQL_DIGESTS` function on the `ALL_SQL_DIGESTS` column of the `TIDB_TRX` table, and converts the SQL digest array into an array of normalized SQL statement through the system internal query. This helps you visually obtain the information of the statements that have been historically executed by the transaction. However, note that the above query scans the entire table of `TIDB_TRX` and calls the `TIDB_DECODE_SQL_DIGESTS` function for each row. Calling the `TIDB_DECODE_SQL_DIGESTS` function has a high overhead. Therefore, if many concurrent transactions exist in the cluster, try to avoid this type of query.

CLUSTER_TIDB_TRX

The `TIDB_TRX` table only provides information about the transactions that are being executed on a single TiDB node. If you want to view the information of the transactions that are being executed on all TiDB nodes in the entire cluster, you need to query the `CLUSTER_TIDB_TRX` table. Compared with the query result of the `TIDB_TRX` table, the query result of the `CLUSTER_TIDB_TRX` table includes an extra `INSTANCE` field. The `INSTANCE` field displays the IP address and port of each node in the cluster, which is used to distinguish the TiDB nodes where the transactions are located.

```
USE information_schema;
DESC cluster_tidb_trx;
```

```
mysql> desc cluster_tidb_trx;
+-----+
| Field          | Type           | Null | Key |
| Default        | Extra          |       |       |
+-----+
| INSTANCE        | varchar(64)    | YES  |       | NULL  |
| ID              | bigint(21) unsigned | NO   | PRI  | NULL  |
| START_TIME      | timestamp(6)   | YES  |       | NULL  |
+-----+
```

CURRENT_SQL_DIGEST varchar(64)						
↳					YES	NULL
↳						
CURRENT_SQL_DIGEST_TEXT text					YES	NULL
↳						
↳						
STATE enum('Idle','Running','LockWaiting','Committing','RollingBack')	YES	NULL				
WAITING_START_TIME timestamp(6)					YES	NULL
↳						
↳						
MEM_BUFFER_KEYS bigint(64)					YES	NULL
↳						
↳						
MEM_BUFFER_BYTES bigint(64)					YES	NULL
↳						
↳						
SESSION_ID bigint(21) unsigned					YES	NULL
↳						
USER varchar(16)					YES	NULL
↳						
↳						
DB varchar(64)					YES	NULL
↳						
↳						
ALL_SQL_DIGESTS text					YES	NULL
↳						
↳						
+--						
↳						
↳						

11.5.17.2.42 TIFLASH_REPLICA

The TIFLASH_REPLICA table provides information about TiFlash replicas available.

```
USE information_schema;
DESC tiflash_replica;
```

Field	Type	Null	Key	Default	Extra
TABLE_SCHEMA	varchar(64)	YES		NULL	
TABLE_NAME	varchar(64)	YES		NULL	
TABLE_ID	bigint(21)	YES		NULL	

REPLIC_A_COUNT bigint(64) YES NULL
LOCATION_LABELS varchar(64) YES NULL
AVAILABLE tinyint(1) YES NULL
PROGRESS double YES NULL
+-----+-----+-----+-----+-----+
7 rows in set (0.01 sec)

11.5.17.2.43 TIKV_REGION_PEERS

The TIKV_REGION_PEERS table shows detailed information of a single Region node in TiKV, such as whether it is a learner or leader.

```
USE information_schema;
DESC tikv_region_peers;
```

Field	Type	Null	Key	Default	Extra
REGION_ID bigint(21) YES NULL					
PEER_ID bigint(21) YES NULL					
STORE_ID bigint(21) YES NULL					
IS_LEARNER tinyint(1) NO 0					
IS_LEADER tinyint(1) NO 0					
STATUS varchar(10) YES 0					
DOWN_SECONDS bigint(21) YES 0					
+-----+-----+-----+-----+-----+					
7 rows in set (0.01 sec)					

For example, you can query the specific TiKV addresses for the top 3 Regions with the maximum value of WRITTEN_BYTES using the following SQL statement:

```
SELECT
    address,
    tikv.address,
    region.region_id
FROM
    tikv_store_status tikv,
    tikv_region_peers peer,
    (SELECT * FROM tikv_region_status ORDER BY written_bytes DESC LIMIT 3)
    ↪ region
WHERE
    region.region_id = peer.region_id
    AND peer.is_leader = 1
    AND peer.store_id = tikv.store_id;
```

Fields in the TIKV_REGION_PEERS table are described as follows:

- REGION_ID: The Region ID.
- PEER_ID: The ID of the Region peer.
- STORE_ID: The ID of the TiKV store where the Region is located.
- IS_LEARNER: Whether the peer is learner.
- IS_LEADER: Whether the peer is leader.
- STATUS: The statuses of a peer:
 - PENDING: Temporarily unavailable.
 - DOWN: Offline and converted. This peer no longer provides service.
 - NORMAL: Running normally.
- DOWN_SECONDS: The duration of being offline, in seconds.

11.5.17.2.44 TIKV_REGION_STATUS

The TIKV_REGION_STATUS table shows some basic information of TiKV Regions via PD's API, like the Region ID, starting and ending key-values, and read and write traffic.

```
USE information_schema;
DESC tikv_region_status;
```

Field	Type	Null	Key	Default	Extra
REGION_ID	bigint(21)	YES		NULL	
START_KEY	text	YES		NULL	
END_KEY	text	YES		NULL	
TABLE_ID	bigint(21)	YES		NULL	
DB_NAME	varchar(64)	YES		NULL	
TABLE_NAME	varchar(64)	YES		NULL	
IS_INDEX	tinyint(1)	NO		0	
INDEX_ID	bigint(21)	YES		NULL	
INDEX_NAME	varchar(64)	YES		NULL	
EPOCH_CONF_VER	bigint(21)	YES		NULL	
EPOCH_VERSION	bigint(21)	YES		NULL	
WRITTEN_BYTES	bigint(21)	YES		NULL	
READ_BYTES	bigint(21)	YES		NULL	
APPROXIMATE_SIZE	bigint(21)	YES		NULL	
APPROXIMATE_KEYS	bigint(21)	YES		NULL	
REPLICATIONSTATUS_STATE	varchar(64)	YES		NULL	
REPLICATIONSTATUS_STATEID	bigint(21)	YES		NULL	

```
+---+
→ -----+-----+-----+-----+
→
17 rows in set (0.00 sec)
```

The descriptions of the columns in the TIKV_REGION_STATUS table are as follows:

- REGION_ID: The ID of the Region.
- START_KEY: The value of the start key of the Region.
- END_KEY: The value of the end key of the Region.
- TABLE_ID: The ID of the table to which the Region belongs.
- DB_NAME: The name of the database to which TABLE_ID belongs.
- TABLE_NAME: The name of the table to which the Region belongs.
- IS_INDEX: Whether the Region data is an index. 0 means that it is not an index, while 1 means that it is an index. If the current Region contains both table data and index data, there will be multiple rows of records, and IS_INDEX is 0 and 1 respectively.
- INDEX_ID: The ID of the index to which the Region belongs. If IS_INDEX is 0, the value of this column is NULL.
- INDEX_NAME: The name of the index to which the Region belongs. If IS_INDEX is 0, the value of this column is NULL.
- EPOCH_CONF_VER: The version number of the Region configuration. The version number increases when a peer is added or removed.
- EPOCH_VERSION: The current version number of the Region. The version number increases when the Region is split or merged.
- WRITTEN_BYTES: The amount of data (bytes) written to the Region.
- READ_BYTES: The amount of data (bytes) that has been read from the Region.
- APPROXIMATE_SIZE: The approximate data size (MB) of the Region.
- APPROXIMATE_KEYS: The approximate number of keys in the Region.
- REPLICATIONSTATUS_STATE: The current replication status of the Region. The status might be UNKNOWN, SIMPLE_MAJORITY, or INTEGRITY_OVER_LABEL.
- REPLICATIONSTATUS_STATEID: The identifier corresponding to REPLICATIONSTATUS_STATE

→ .

Also, you can implement the `top confver`, `top read` and `top write` operations in pd-ctl via the `ORDER BY X LIMIT Y` operation on the EPOCH_CONF_VER, WRITTEN_BYTES and READ_BYTES columns.

You can query the top 3 Regions with the most write data using the following SQL statement:

```
SELECT * FROM tikv_region_status ORDER BY written_bytes DESC LIMIT 3;
```

11.5.17.2.45 TIKV_STORE_STATUS

The TIKV_STORE_STATUS table shows some basic information of TiKV nodes via PD's API, like the ID allocated in the cluster, address and port, and status, capacity, and the number of Region leaders of the current node.

```
USE information_schema;
DESC tikv_store_status;
```

Field	Type	Null	Key	Default	Extra
STORE_ID	bigint(21)	YES		NULL	
ADDRESS	varchar(64)	YES		NULL	
STORE_STATE	bigint(21)	YES		NULL	
STORE_STATE_NAME	varchar(64)	YES		NULL	
LABEL	json	YES		NULL	
VERSION	varchar(64)	YES		NULL	
CAPACITY	varchar(64)	YES		NULL	
AVAILABLE	varchar(64)	YES		NULL	
LEADER_COUNT	bigint(21)	YES		NULL	
LEADER_WEIGHT	double	YES		NULL	
LEADER_SCORE	double	YES		NULL	
LEADER_SIZE	bigint(21)	YES		NULL	
REGION_COUNT	bigint(21)	YES		NULL	
REGION_WEIGHT	double	YES		NULL	
REGION_SCORE	double	YES		NULL	
REGION_SIZE	bigint(21)	YES		NULL	
START_TS	datetime	YES		NULL	
LAST_HEARTBEAT_TS	datetime	YES		NULL	
UPTIME	varchar(64)	YES		NULL	

19 rows in set (0.00 sec)

The descriptions of the columns in the TIKV_STORE_STATUS table are as follows:

- STORE_ID: The ID of the Store.
- ADDRESS: The address of the Store.
- STORE_STATE: The identifier of the Store state, which corresponds to STORE_STATE_NAME ↪ .
- STORE_STATE_NAME: The name of the Store state. The name is Up, Offline, or Tombstone.
- LABEL: The label set for the Store.
- VERSION: The version number of the Store.
- CAPACITY: The storage capacity of the Store.
- AVAILABLE: The remaining storage space of the Store.
- LEADER_COUNT: The number of leaders on the Store.

- LEADER_WEIGHT: The leader weight of the Store.
- LEADER_SCORE: The leader score of the Store.
- LEADER_SIZE: The approximate total data size (MB) of all leaders on the Store.
- REGION_COUNT: The number of Regions on the Store.
- REGION_WEIGHT: The Region weight of the Store.
- REGION_SCORE: The Region score of the Store.
- REGION_SIZE: The approximate total data size (MB) of all Regions on the Store.
- START_TS: The timestamp when the Store is started.
- LAST_HEARTBEAT_TS: The timestamp of the last heartbeat sent by the Store.
- UPTIME: The total time since the Store starts.

11.5.17.2.46 USER_PRIVILEGES

The `USER_PRIVILEGES` table provides information about global privileges. This information comes from the `mysql.user` system table:

```
USE information_schema;
DESC user_privileges;
```

Field	Type	Null	Key	Default	Extra
GRANTEE	varchar(81)	YES		NULL	
TABLE_CATALOG	varchar(512)	YES		NULL	
PRIVILEGE_TYPE	varchar(64)	YES		NULL	
IS_GRANTABLE	varchar(3)	YES		NULL	

4 rows in set (0.00 sec)

```
SELECT * FROM user_privileges;
```

GRANTEE	TABLE_CATALOG	PRIVILEGE_TYPE	IS_GRANTABLE
'root'@'%'	def	Select	YES
'root'@'%'	def	Insert	YES
'root'@'%'	def	Update	YES
'root'@'%'	def	Delete	YES
'root'@'%'	def	Create	YES
'root'@'%'	def	Drop	YES
'root'@'%'	def	Process	YES
'root'@'%'	def	References	YES
'root'@'%'	def	Alter	YES
'root'@'%'	def	Show Databases	YES
'root'@'%'	def	Super	YES

'root'@'%'	def	Execute	YES	
'root'@'%'	def	Index	YES	
'root'@'%'	def	Create User	YES	
'root'@'%'	def	Trigger	YES	
'root'@'%'	def	Create View	YES	
'root'@'%'	def	Show View	YES	
'root'@'%'	def	Create Role	YES	
'root'@'%'	def	Drop Role	YES	
'root'@'%'	def	CREATE TEMPORARY TABLES	YES	
'root'@'%'	def	LOCK TABLES	YES	
'root'@'%'	def	CREATE ROUTINE	YES	
'root'@'%'	def	ALTER ROUTINE	YES	
'root'@'%'	def	EVENT	YES	
'root'@'%'	def	SHUTDOWN	YES	
'root'@'%'	def	RELOAD	YES	
'root'@'%'	def	FILE	YES	
'root'@'%'	def	CONFIG	YES	

28 rows in set (0.00 sec)

Fields in the `USER_PRIVILEGES` table are described as follows:

- **GRANTEE**: The name of the granted user, which is in the format of '`user_name`'@'`host_name`'.
- **TABLE_CATALOG**: The name of the catalog to which the table belongs. This value is always `def`.
- **PRIVILEGE_TYPE**: The privilege type to be granted. Only one privilege type is shown in each row.
- **IS_GRANTABLE**: If you have the `GRANT OPTION` privilege, the value is `YES`; otherwise, the value is `NO`.

11.5.17.2.47 VIEWS

The `VIEWS` table provides information about SQL views.

```
USE information_schema;
DESC views;
```

Field	Type	Null	Key	Default	Extra
TABLE_CATALOG	varchar(512)	NO		NULL	
TABLE_SCHEMA	varchar(64)	NO		NULL	
TABLE_NAME	varchar(64)	NO		NULL	
VIEW_DEFINITION	longblob	NO		NULL	

CHECK_OPTION	varchar(8)	NO	NULL		
IS_UPDATABLE	varchar(3)	NO	NULL		
DEFINER	varchar(77)	NO	NULL		
SECURITY_TYPE	varchar(7)	NO	NULL		
CHARACTER_SET_CLIENT	varchar(32)	NO	NULL		
COLLATION_CONNECTION	varchar(32)	NO	NULL		

10 rows in set (0.00 sec)

```
CREATE VIEW test.v1 AS SELECT 1;
SELECT * FROM views\G
```

```
***** 1. row *****
TABLE_CATALOG: def
TABLE_SCHEMA: test
TABLE_NAME: v1
VIEW_DEFINITION: SELECT 1
CHECK_OPTION: CASCDED
IS_UPDATABLE: NO
DEFINER: root@127.0.0.1
SECURITY_TYPE: DEFINER
CHARACTER_SET_CLIENT: utf8mb4
COLLATION_CONNECTION: utf8mb4_0900_ai_ci
1 row in set (0.00 sec)
```

Fields in the VIEWS table are described as follows:

- **TABLE_CATALOG**: The name of the catalog to which the view belongs. This value is always `def`.
- **TABLE_SCHEMA**: The name of the schema to which the view belongs.
- **TABLE_NAME**: The view name.
- **VIEW_DEFINITION**: The definition of view, which is made by the `SELECT` statement when the view is created.
- **CHECK_OPTION**: The `CHECK_OPTION` value. The value options are `NONE`, `CASCADE`, and `LOCAL`.
- **IS_UPDATABLE**: Whether `UPDATE/INSERT/DELETE` is applicable to the view. In TiDB, the value is always `NO`.
- **DEFINER**: The name of the user who creates the view, which is in the format of '`→ user_name '@' host_name'`.
- **SECURITY_TYPE**: The value of `SQL SECURITY`. The value options are `DEFINER` and `INVOKER`.
- **CHARACTER_SET_CLIENT**: The value of the `character_set_client` session variable when the view is created.
- **COLLATION_CONNECTION**: The value of the `collation_connection` session variable when the view is created.

11.5.17.3 Metrics Schema

The METRICS_SCHEMA is a set of views on top of TiDB metrics that are stored in Prometheus. The source of the PromQL (Prometheus Query Language) for each of the tables is available in `INFORMATION_SCHEMA.METRICS_TABLES`.

```
USE metrics_schema;
SELECT * FROM uptime;
SELECT * FROM information_schema.metrics_tables WHERE table_name='uptime' \G
```

time	instance	job	value
2020-07-06 15:26:26.203000	127.0.0.1:10080	tidb	123.60300016403198
2020-07-06 15:27:26.203000	127.0.0.1:10080	tidb	183.60300016403198
2020-07-06 15:26:26.203000	127.0.0.1:20180	tikv	123.60300016403198
2020-07-06 15:27:26.203000	127.0.0.1:20180	tikv	183.60300016403198
2020-07-06 15:26:26.203000	127.0.0.1:2379	pd	123.60300016403198
2020-07-06 15:27:26.203000	127.0.0.1:2379	pd	183.60300016403198
2020-07-06 15:26:26.203000	127.0.0.1:9090	prometheus	123.72300004959106
2020-07-06 15:27:26.203000	127.0.0.1:9090	prometheus	183.72300004959106

8 rows in set (0.00 sec)

```
***** 1. row *****
TABLE_NAME: uptime
PROMQL: (time() - process_start_time_seconds{$LABEL_CONDITIONS})
LABELS: instance,job
QUANTILE: 0
COMMENT: TiDB uptime since last restart(second)
1 row in set (0.00 sec)
```

```
SHOW TABLES;
```

Tables_in_metrics_schema
abnormal_stores

```

| etcd_disk_wal_fsync_rate           | |
| etcd_wal_fsync_duration           | |
| etcd_wal_fsync_total_count        | |
| etcd_wal_fsync_total_time         | |
| go_gc_count                      | |
| go_gc_cpu_usage                  | |
| go_gc_duration                   | |
| go_heap_mem_usage                | |
| go_threads                       | |
| goroutines_count                 | |
| node_cpu_usage                   | |
| node_disk_available_size         | |
| node_disk_io_util                | |
| node_disk_iops                   | |
| node_disk_read_latency           | |
| node_disk_size                   | |
| ..                                | |
| tikv_storage_async_request_total_time | |
| tikv_storage_async_requests       | |
| tikv_storage_async_requests_total_count | |
| tikv_storage_command_ops          | |
| tikv_store_size                  | |
| tikv_thread_cpu                  | |
| tikv_thread_nonvoluntary_context_switches | |
| tikv_thread_voluntary_context_switches | |
| tikv_threads_io                  | |
| tikv_threads_state                | |
| tikv_total_keys                  | |
| tikv_wal_sync_duration           | |
| tikv_wal_sync_max_duration       | |
| tikv_worker_handled_tasks        | |
| tikv_worker_handled_tasks_total_num | |
| tikv_worker_pending_tasks         | |
| tikv_worker_pending_tasks_total_num | |
| tikv_write_stall_avg_duration    | |
| tikv_write_stall_max_duration    | |
| tikv_write_stall_reason          | |
| up                               | |
| uptime                           | |
+-----+
626 rows in set (0.00 sec)

```

The METRICS_SCHEMA is used as a data source for monitoring-related summary tables such as ([metrics_summary](#), [metrics_summary_by_label](#) and [inspection_summary](#).

11.5.17.3.1 Additional Examples

Taking the `tidb_query_duration` monitoring table in `metrics_schema` as an example, this section illustrates how to use this monitoring table and how it works. The working principles of other monitoring tables are similar to `tidb_query_duration`.

Query the information related to the `tidb_query_duration` table on `information_schema`

```
↪ .metrics_tables:
```

```
SELECT * FROM information_schema.metrics_tables WHERE table_name='
↪ tidb_query_duration';
```

TABLE_NAME	PROMQL
LABELS	QUANTILE COMMENT
tidb_query_duration	histogram_quantile(\$QUANTILE, sum(rate(tidb_server_handle_query_duration_seconds_bucket{\$LABEL_CONDITIONS}[\$RANGE_DURATION]) by (le,sql_type,instance)) instance,sql_type 0.9 The quantile of TiDB query durations(second)

Field description:

- **TABLE_NAME**: Corresponds to the table name in `metrics_schema`. In this example, the table name is `tidb_query_duration`.
- **PROMQL**: The working principle of the monitoring table is to first map SQL statements to PromQL, then to request data from Prometheus, and to convert Prometheus results into SQL query results. This field is the expression template of PromQL. When you query the data of the monitoring table, the query conditions are used to rewrite the variables in this template to generate the final query expression.
- **LABELS**: The label for the monitoring item. `tidb_query_duration` has two labels: `instance` and `sql_type`.
- **QUANTILE**: The percentile. For monitoring data of the histogram type, a default percentile is specified. If the value of this field is 0, it means that the monitoring item corresponding to the monitoring table is not a histogram.
- **COMMENT**: Explanations for the monitoring table. You can see that the `tidb_query_duration` table is used to query the percentile time of the TiDB query execution, such as the query time of P999/P99/P90. The unit is second.

To query the schema of the `tidb_query_duration` table, execute the following statement:

```
SHOW CREATE TABLE metrics_schema.tidb_query_duration;
```

```
+--+
→ +-----+ | Create Table
→ | Table      | +-----+
→ |           | |
→ |           | |
+--+
→ +-----+ | |
→ | tidb_query_duration | CREATE TABLE `tidb_query_duration` (
→ |           |   `time` datetime unsigned DEFAULT CURRENT_TIMESTAMP,
→ |           |   |
→ |           |   `instance` varchar(512) DEFAULT NULL,
→ |           |   |
→ |           |   `sql_type` varchar(512) DEFAULT NULL,
→ |           |   |
→ |           |   `quantile` double unsigned DEFAULT '0.9',
→ |           |   |
→ |           |   `value` double unsigned DEFAULT NULL
→ |           |   |
→ |           |   ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=
→ |           |   utf8mb4_bin COMMENT='The quantile of TiDB query durations(second)' |
+--+
→ +-----+
```

- `time`: The time of the monitoring item.
- `instance` and `sql_type`: The labels of the `tidb_query_duration` monitoring item. `instance` means the monitoring address. `sql_type` means the type of the executed SQL statement.
- `quantile`: The percentile. The monitoring item of the histogram type has this column, which indicates the percentile time of the query. For example, `quantile = 0.9` means to query the time of P90.
- `value`: The value of the monitoring item.

The following statement queries the P99 time within the range of [2020-03-25 23:40:00 → , 2020-03-25 23:42:00].

```
SELECT * FROM metrics_schema.tidb_query_duration WHERE value is not null
  ↵ AND time>='2020-03-25 23:40:00' AND time <= '2020-03-25 23:42:00' AND
  ↵ quantile=0.99;
```

time	instance	sql_type	quantile	value
2020-03-25 23:40:00	172.16.5.40:10089	Insert	0.99	0.509929485256
2020-03-25 23:41:00	172.16.5.40:10089	Insert	0.99	0.494690793986
2020-03-25 23:42:00	172.16.5.40:10089	Insert	0.99	0.493460506934
2020-03-25 23:40:00	172.16.5.40:10089	Select	0.99	0.152058493415
2020-03-25 23:41:00	172.16.5.40:10089	Select	0.99	0.152193879678
2020-03-25 23:42:00	172.16.5.40:10089	Select	0.99	0.140498483232
2020-03-25 23:40:00	172.16.5.40:10089	internal	0.99	0.47104
2020-03-25 23:41:00	172.16.5.40:10089	internal	0.99	0.11776
2020-03-25 23:42:00	172.16.5.40:10089	internal	0.99	0.11776

The first row of the query result above means that at the time of 2020-03-25 23:40:00, on the TiDB instance 172.16.5.40:10089, the P99 execution time of the `Insert` type statement is 0.509929485256 seconds. The meanings of other rows are similar. Other values of the `sql_type` column are described as follows:

- `Select`: The `select` type statement is executed.
- `internal`: The internal SQL statement of TiDB, which is used to update the statistical information and get the global variables.

To view the execution plan of the statement above, execute the following statement:

```
DESC SELECT * FROM metrics_schema.tidb_query_duration WHERE value is not
  ↵ null AND time>='2020-03-25 23:40:00' AND time <= '2020-03-25 23:42:00'
  ↵ AND quantile=0.99;
```

+--					
→	→	→	→	→	→
id	estRows	task	access object	operator info	
→					
→					
+--					
→	→	→	→	→	→
Selection_5	8000.00	root		not(isnull(Column	
→ #5))					
→					
→					
-MemTableScan_6	10000.00	root	table:tidb_query_duration	PromQL:	
→ histogram_quantile(0.99, sum(rate(
→ tidb_server_handle_query_duration_seconds_bucket{}[60s]))			by (le,		
→ sql_type,instance)), start_time:2020-03-25 23:40:00, end_time					
→ :2020-03-25 23:42:00, step:1m0s					
+--					
→	→	→	→	→	→

From the result above, you can see that `PromQL`, `start_time`, `end_time`, and `step` are in the execution plan. During the execution process, TiDB calls the `query_range` HTTP API of Prometheus to query the monitoring data.

You might find that in the range of [2020-03-25 23:40:00, 2020-03-25 23:42:00], each label only has three time values. In the execution plan, the value of `step` is 1 minute, which means that the interval of these values is 1 minute. `step` is determined by the following two session variables:

- `tidb_metric_query_step`: The query resolution step width. To get the `query_range` data from Prometheus, you need to specify `start_time`, `end_time`, and `step`. `step` uses the value of this variable.
- `tidb_metric_query_range_duration`: When the monitoring data is queried, the value of the `$ RANGE_DURATION` field in PROMQL is replaced with the value of this variable. The default value is 60 seconds.

To view the values of monitoring items with different granularities, you can modify the two session variables above before querying the monitoring table. For example:

1. Modify the values of the two session variables and set the time granularity to 30 seconds.

Note:

The minimum granularity supported by Prometheus is 30 seconds.

```
set @@tidb_metric_query_step=30;
set @@tidb_metric_query_range_duration=30;
```

2. Query the `tidb_query_duration` monitoring item as follows. From the result, you can see that within the 3-minute time range, each label has 6 time values, and the interval between each value is 30 seconds.

```
select * from metrics_schema.tidb_query_duration where value is not
→ null and time>='2020-03-25 23:40:00' and time <= '2020-03-25
→ 23:42:00' and quantile=0.99;
```

time	instance	sql_type	quantile	value
2020-03-25 23:40:00	172.16.5.40:10089	Insert	0.99	0.483285651924
2020-03-25 23:40:30	172.16.5.40:10089	Insert	0.99	0.484151462113
2020-03-25 23:41:00	172.16.5.40:10089	Insert	0.99	0.504576
2020-03-25 23:41:30	172.16.5.40:10089	Insert	0.99	0.493577384561
2020-03-25 23:42:00	172.16.5.40:10089	Insert	0.99	0.49482474311
2020-03-25 23:40:00	172.16.5.40:10089	Select	0.99	0.189253402185
2020-03-25 23:40:30	172.16.5.40:10089	Select	0.99	0.184224951851
2020-03-25 23:41:00	172.16.5.40:10089	Select	0.99	0.151673410553
2020-03-25 23:41:30	172.16.5.40:10089	Select	0.99	0.127953838989
2020-03-25 23:42:00	172.16.5.40:10089	Select	0.99	0.127455434547

2020-03-25 23:40:00 172.16.5.40:10089 internal 0.99 0.0624
↳
2020-03-25 23:40:30 172.16.5.40:10089 internal 0.99 0.12416
↳
2020-03-25 23:41:00 172.16.5.40:10089 internal 0.99 0.0304
↳
2020-03-25 23:41:30 172.16.5.40:10089 internal 0.99 0.06272
↳
2020-03-25 23:42:00 172.16.5.40:10089 internal 0.99
↳ 0.0629333333333333
+--
↳ -----+-----+-----+-----+
↳

3. View the execution plan. From the result, you can also see that the values of PromQL and step in the execution plan have been changed to 30 seconds.

```
desc select * from metrics_schema.tidb_query_duration where value is
  ↳ not null and time>='2020-03-25 23:40:00' and time <= '2020-03-25
  ↳ 23:42:00' and quantile=0.99;
```

+--
↳ -----+-----+-----+-----+
↳
id estRows task access object operator
↳ info
↳
↳
+--
↳ -----+-----+-----+-----+
↳
Selection_5 8000.00 root not(isnull(↳ Column#5))
↳
↳
-MemTableScan_6 10000.00 root table:tidb_query_duration ↳ PromQL:histogram_quantile(0.99, sum(rate(↳ tidb_server_handle_query_duration_seconds_bucket{}[30s])) by (le ↳ ,sql_type,instance)), start_time:2020-03-25 23:40:00, end_time ↳ :2020-03-25 23:42:00, step:30s
+--
↳ -----+-----+-----+-----+
↳

11.6 UI

11.6.1 TiDB Dashboard

11.6.1.1 TiDB Dashboard Introduction

TiDB Dashboard is a Web UI for monitoring, diagnosing, and managing the TiDB cluster, which is available since v4.0. It is built into the PD component and does not require an independent deployment.

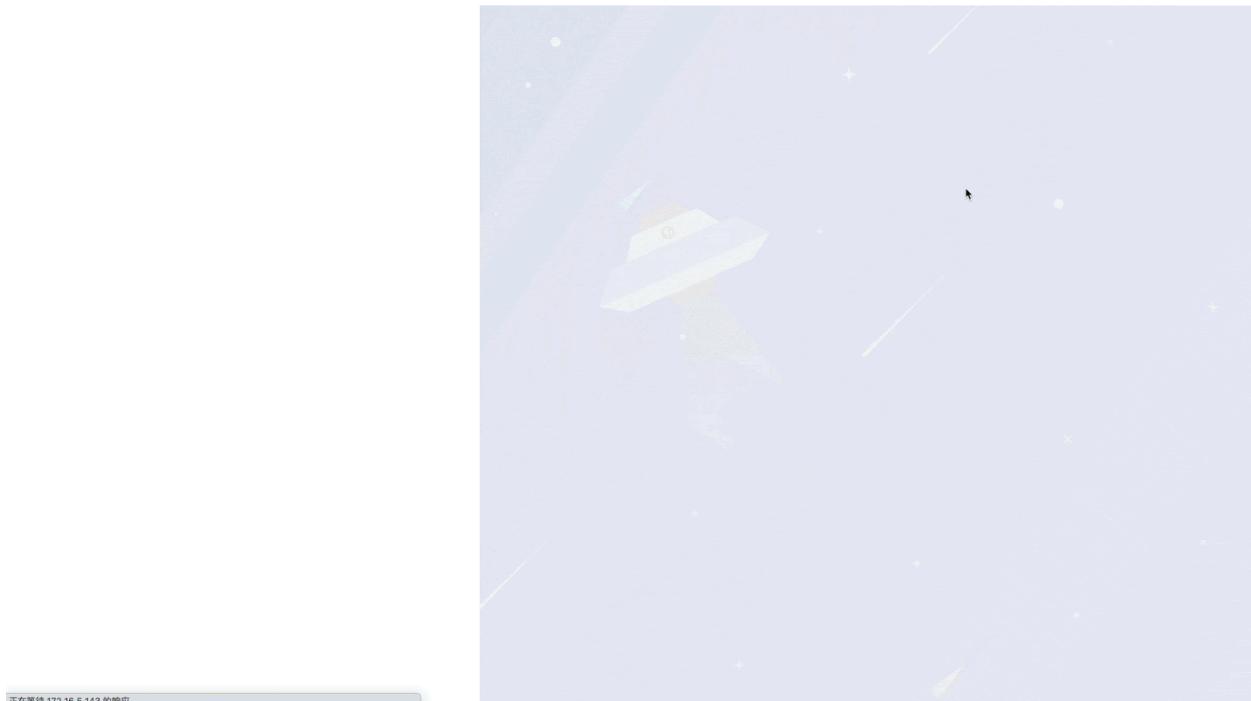


Figure 337: TiDB Dashboard interface

TiDB Dashboard is open-sourced on [GitHub](#).

This document introduces the main features of TiDB Dashboard. You can click links in the following sections to learn more details.

11.6.1.1.1 Show the overall running status of the TiDB cluster

You can use TiDB Dashboard to learn the TiDB cluster's queries per second (QPS), execution time, the types of SQL statements that consume the most resources, and other overview information.

See [TiDB Dashboard Overview](#) for details.

11.6.1.1.2 Show the running status of components and hosts

You can use TiDB Dashboard to view the running status of TiDB, TiKV, PD, TiFlash components in the entire cluster and the running status of the host on which these components are located.

See [TiDB Dashboard Cluster Information Page](#) for details.

11.6.1.1.3 Show distribution and trends of read and write traffic

The Key Visualizer feature of TiDB Dashboard visually shows the change of read and write traffic over time in the entire cluster in the form of heatmap. You can use this feature to timely discover changes of application modes or locate hotspot issues with uneven performance.

See [Key Visualizer Page](#) for details.

11.6.1.1.4 Show a list of execution information of all SQL statements

The execution information of all SQL statements is listed on the SQL Statements page. You can use this page to learn the execution time and total executions at all stages, which helps you analyze and locate the SQL queries that consume the most resources and improve the overall cluster performance.

See [SQL Statements Page of TiDB Dashboard](#) for details.

11.6.1.1.5 Learn the detailed execution information of slow queries

The Slow Queries page of TiDB Dashboard shows a list of all SQL statements that take a long time to execute, including the SQL texts and execution information. This page helps you locate the cause of slow queries or performance jitter.

See [Slow Queries Page](#) for details.

11.6.1.1.6 Diagnose common cluster problems and generate reports

The diagnostic feature of TiDB Dashboard automatically determines whether some common risks (such as inconsistent configurations) or problems exist in the cluster, generates reports, and gives operation suggestions, or compares the status of each cluster metric in different time ranges for you to analyze possible problems.

See [TiDB Dashboard Cluster Diagnostics Page](#) for details.

11.6.1.1.7 Query logs of all components

On the Search Logs page of TiDB Dashboard, you can quickly search logs of all running instances in the cluster by keywords, time range, and other conditions, package these logs, and download them to your local machine.

See [Search Logs Page](#) for details.

11.6.1.1.8 Collect profiling data for each instance

This is an advanced debugging feature that lets you profile each instance online and analyze various internal operations an instance performed during the profiling data collection period and the proportion of the operation execution time in this period without third-party tools.

See [Profile Instances Page](#) for details.

Note:

By default, TiDB Dashboard shares usage details with PingCAP to help understand how to improve the product. For details about what is shared and how to disable the sharing, see [Telemetry](#).

11.6.1.2 Maintain

11.6.1.2.1 Deploy TiDB Dashboard

The TiDB Dashboard UI is built into the PD component for v4.0 or higher versions, and no additional deployment is required. Simply deploy a standard TiDB cluster, and TiDB Dashboard will be there.

See the following documents to learn how to deploy a standard TiDB cluster:

- [Quick Start Guide for the TiDB Database Platform](#)
- [Deploy TiDB in Production Environment](#)
- [Kubernetes environment deployment](#)

Note:

You cannot deploy TiDB Dashboard in a TiDB cluster earlier than v4.0.

Deployment with multiple PD instances

When multiple PD instances are deployed in the cluster, only one of these instances serves the TiDB Dashboard.

When PD instances are running for the first time, they automatically negotiate with each other to choose one instance to serve the TiDB Dashboard. TiDB Dashboard will not run on other PD instances. The TiDB Dashboard service will always be provided by the chosen PD instance no matter PD instances are restarted or new PD instances are joined. However,

there will be a re-negotiation when the PD instance that serves TiDB Dashboard is removed from the cluster (scaled-in). The negotiation process does not need user intervention.

When you access a PD instance that does not serve TiDB Dashboard, the browser will be redirected automatically to guide you to access the PD instance that serves the TiDB Dashboard, so that you can access the service normally. This process is illustrated in the image below.

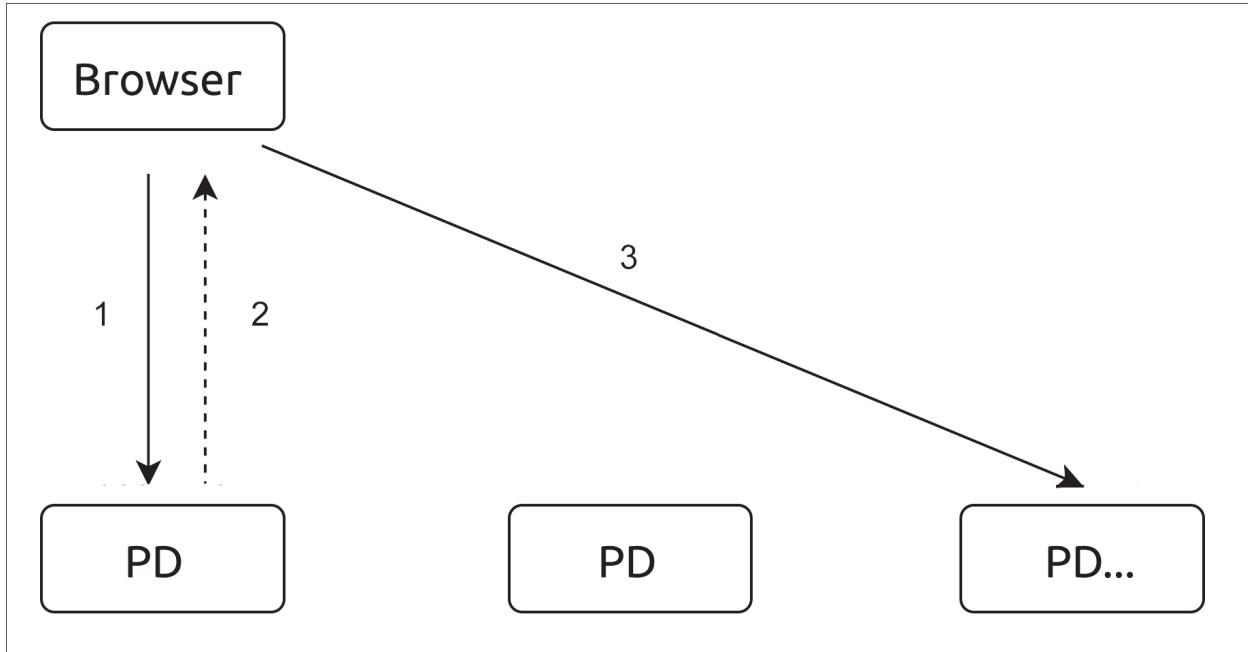


Figure 338: Process Schematic

Note:

The PD instance that serves TiDB Dashboard might not be a PD leader.

Check the PD instance that actually serves TiDB Dashboard

For a running cluster deployed using TiUP, you can use the `tiup cluster display` command to see which PD instance serves TiDB Dashboard. Replace `CLUSTER_NAME` with the cluster name.

```
tiup cluster display CLUSTER_NAME --dashboard
```

A sample output is as follows:

```
http://192.168.0.123:2379/dashboard/
```

Note:

This feature is available only in the later version of the `tiup cluster` deployment tool (v1.0.3 or later).

Upgrade TiUP Cluster

```
tiup update --self  
tiup update cluster --force
```

Switch to another PD instance to serve TiDB Dashboard

For a running cluster deployed using TiUP, you can use the `tiup ctl pd` command to change the PD instance that serves TiDB Dashboard, or re-specify a PD instance to serve TiDB Dashboard when it is disabled:

```
tiup ctl pd -u http://127.0.0.1:2379 config set dashboard-address http  
↪ ://9.9.9.9:2379
```

In the command above:

- Replace `127.0.0.1:2379` with the IP and port of any PD instance.
- Replace `9.9.9.9:2379` with the IP and port of the new PD instance that you desire to run the TiDB Dashboard service.

You can use the `tiup cluster display` command to see whether the modification is taking effect (replace `CLUSTER_NAME` with the cluster name):

```
tiup cluster display CLUSTER_NAME --dashboard
```

Warning:

If you change the instance to run TiDB Dashboard, the local data stored in the previous TiDB Dashboard instance will be lost, including the Key Visualize history and search history.

Disable TiDB Dashboard

For a running cluster deployed using TiUP, use the `tiup ctl pd` command to disable TiDB Dashboard on all PD instances (replace `127.0.0.1:2379` with the IP and port of any PD instance):

```
tiup ctl pd -u http://127.0.0.1:2379 config set dashboard-address none
```

After disabling TiDB Dashboard, checking which PD instance provides the TiDB Dashboard service will fail:

```
Error: TiDB Dashboard is disabled
```

Visiting the TiDB Dashboard address of any PD instance via the browser will also fail:

```
Dashboard is not started.
```

Re-enable TiDB Dashboard

For a running cluster deployed using TiUP, use the `tiup ctl pd` command to request PD to renegotiate an instance to run TiDB Dashboard (replace `127.0.0.1:2379` with the IP and port of any PD instance):

```
tiup ctl pd -u http://127.0.0.1:2379 config set dashboard-address auto
```

After executing the command above, you can use the `tiup cluster display` command to view the TiDB Dashboard instance address automatically negotiated by PD (replace `CLUSTER_NAME` with the cluster name):

```
tiup cluster display CLUSTER_NAME --dashboard
```

You can also re-enable TiDB Dashboard by manually specifying the PD instance that serves TiDB Dashboard. See [Switch to another PD instance to serve TiDB Dashboard](#).

Warning:

If the newly enabled TiDB Dashboard instance is different with the previous instance that served the TiDB Dashboard, the local data stored in the previous TiDB Dashboard instance will be lost, including Key Visualize history and search history.

What's next

- To learn how to access and log into the TiDB Dashboard UI, see [Access TiDB Dashboard](#).
- To learn how to enhance the security of TiDB Dashboard, such as configuring a firewall, see [Secure TiDB Dashboard](#).

11.6.1.2.2 Use TiDB Dashboard behind a Reverse Proxy

You can use a reverse proxy to safely expose the TiDB Dashboard service from the internal network to the external.

Procedures

Step 1: Get the actual TiDB Dashboard address

When multiple PD instances are deployed in the cluster, only one of the PD instances actually runs TiDB Dashboard. Therefore, you need to ensure that the upstream of the reverse proxy points to the correct address. For details of this mechanism, see [Deployment with multiple PD instances](#).

When you use the TiUP tool for deployment, execute the following command to get the actual TiDB Dashboard address (replace CLUSTER_NAME with your cluster name):

```
tiup cluster display CLUSTER_NAME --dashboard
```

The output is the actual TiDB Dashboard address. A sample is as follows:

```
http://192.168.0.123:2379/dashboard/
```

Note:

This feature is available only in the later version of the `tiup cluster` deployment tool (v1.0.3 or later).

Upgrade TiUP Cluster

```
tiup update --self  
tiup update cluster --force
```

Step 2: Configure the reverse proxy

Use HAProxy

When you use [HAProxy](#) as the reverse proxy, take the following steps:

1. Use reverse proxy for TiDB Dashboard on the 8033 port (for example). In the HAProxy configuration file, add the following configuration:

```
frontend tidb_dashboard_front  
bind *:8033  
use_backend tidb_dashboard_back if { path /dashboard } or { path_beg  
↪ /dashboard/ }
```

```
backend tidb_dashboard_back
  mode http
  server tidb_dashboard 192.168.0.123:2379
```

Replace 192.168.0.123:2379 with IP and port of the actual address of the TiDB Dashboard obtained in [Step 1](#).

Warning:

You must retain the `if` part in the `use_backend` directive to ensure that services **only in this path** are behind reverse proxy; otherwise, security risks might be introduced. See [Secure TiDB Dashboard](#).

2. Restart HAProxy for the configuration to take effect.
3. Test whether the reverse proxy is effective: access the `/dashboard/` address on the 8033 port of the machine where HAProxy is located (such as `http://example.com ↗ :8033/dashboard/`) to access TiDB Dashboard.

Use NGINX

When you use **NGINX** as the reverse proxy, take the following steps:

1. Use reverse proxy for TiDB Dashboard on the 8033 port (for example). In the NGINX configuration file, add the following configuration:

```
server {
  listen 8033;
  location /dashboard/ {
    proxy_pass http://192.168.0.123:2379/dashboard/;
  }
}
```

Replace `http://192.168.0.123:2379/dashboard/` with the actual address of the TiDB Dashboard obtained in [Step 1](#).

Warning:

You must keep the `/dashboard/` path in the `proxy_pass` directive to ensure that only the services under this path are reverse proxied. Otherwise, security risks will be introduced. See [Secure TiDB Dashboard](#).

2. Reload NGINX for the configuration to take effect.

```
sudo nginx -s reload
```

3. Test whether the reverse proxy is effective: access the `/dashboard/` address on the 8033 port of the machine where NGINX is located (such as `http://example.com:8033/dashboard/`) to access TiDB Dashboard.

Customize path prefix

TiDB Dashboard provides services by default in the `/dashboard/` path, such as `http://example.com:8033/dashboard/`, which is the case even for reverse proxies. To configure the reverse proxy to provide the TiDB Dashboard service with a non-default path, such as `http://example.com:8033/foo/` or `http://example.com:8033/`, take the following steps.

Step 1: Modify PD configuration to specify the path prefix of TiDB Dashboard service

Modify the `public-path-prefix` configuration item in the `[dashboard]` category of the PD configuration to specify the path prefix of the TiDB Dashboard service. After this item is modified, restart the PD instance for the modification to take effect.

For example, if the cluster is deployed using TiUP and you want the service to run on `http://example.com:8033/foo/`, you can specify the following configuration:

```
server_configs:
  pd:
    dashboard.public-path-prefix: /foo
```

Modify configuration when deploying a new cluster using TiUP

If you are deploying a new cluster, you can add the configuration above to the `topology.yaml` TiUP topology file and deploy the cluster. For specific instruction, see [TiUP deployment document](#).

Modify configuration of a deployed cluster using TiUP

For a deployed cluster:

1. Open the configuration file of the cluster in the edit mode (replace `CLUSTER_NAME` with the cluster name).

```
tiup cluster edit-config CLUSTER_NAME
```

2. Modify or add configuration items under the `pd` configuration of `server_configs`. If no `server_configs` exists, add it at the top level:

```
monitored:
  ...
server_configs:
  tidb: ...
  tikv: ...
  pd:
    dashboard.public-path-prefix: /foo
  ...
```

The configuration file after the modification is similar to the following file:

```
server_configs:
  pd:
    dashboard.public-path-prefix: /foo
  global:
    user: tidb
  ...
  ...
```

Or

```
monitored:
  ...
server_configs:
  tidb: ...
  tikv: ...
  pd:
    dashboard.public-path-prefix: /foo
```

3. Perform a rolling restart to all PD instances for the modified configuration to take effect (replace CLUSTER_NAME with your cluster name):

```
shell tiup cluster reload CLUSTER_NAME -R pd
```

See [Common TiUP Operations - Modify the configuration](#) for details.

If you want that the TiDB Dashboard service is run in the root path (such as `http://example.com:8033/`), use the following configuration:

```
server_configs:
  pd:
    dashboard.public-path-prefix: /
```

Warning:

After the modified and customized path prefix takes effect, you cannot directly access TiDB Dashboard. You can only access TiDB Dashboard through a reverse proxy that matches the path prefix.

Step 2: Modify the reverse proxy configuration

Use HAProxy

Taking `http://example.com:8033/foo/` as an example, the corresponding HAProxy configuration is as follows:

```

frontend tidb_dashboard_front
  bind *:8033
  use_backend tidb_dashboard_back if { path /foo } or { path_beg /foo/ }

backend tidb_dashboard_back
  mode http
  http-request set-path %[path,regsub(^/foo/?,/dashboard/)]
  server tidb_dashboard 192.168.0.123:2379

```

Replace 192.168.0.123:2379 with IP and port of the actual address of the TiDB Dashboard obtained in [Step 1](#).

Warning:

You must retain the `if` part in the `use_backend` directive to ensure that services **only in this path** are behind reverse proxy; otherwise, security risks might be introduced. See [Secure TiDB Dashboard](#).

If you want that the TiDB Dashboard service is run in the root path (such as `http://example.com:8033/`), use the following configuration:

```

frontend tidb_dashboard_front
  bind *:8033
  use_backend tidb_dashboard_back
backend tidb_dashboard_back
  mode http
  http-request set-path /dashboard%[path]
  server tidb_dashboard 192.168.0.123:2379

```

Modify the configuration and restart HAProxy for the modified configuration to take effect.

Use NGINX

Taking `http://example.com:8033/foo/` as an example, the corresponding NGINX configuration is as follows:

```

server {
  listen 8033;
  location /foo/ {
    proxy_pass http://192.168.0.123:2379/dashboard/;
  }
}

```

Replace `http://192.168.0.123:2379/dashboard/` with the actual address of the TiDB Dashboard obtained in [Step 1](#).

Warning:

You must retain the `/dashboard/` path in the `proxy_pass` directive to ensure that services **only in this path** are behind reverse proxy; otherwise, security risks might be introduced. See [Secure TiDB Dashboard](#).

If you want that the TiDB Dashboard service is run in the root path (such as `http://example.com:8033/`), use the following configuration:

```
server {  
    listen 8033;  
    location / {  
        proxy_pass http://192.168.0.123:2379/dashboard/;  
    }  
}
```

Modify the configuration and restart NGINX for the modified configuration to take effect.

```
sudo nginx -s reload
```

What's next

To learn how to enhance the security of TiDB Dashboard, such as configuring a firewall, see [Secure TiDB Dashboard](#).

11.6.1.2.3 TiDB Dashboard User Management

TiDB Dashboard uses the same user privilege system and sign-in authentication as TiDB. You can control and manage TiDB SQL users to limit their access to TiDB Dashboard. This document describes the least privileges required for TiDB SQL users to access TiDB Dashboard and exemplifies how to create a least-privileged SQL user.

For details about how to control and manage TiDB SQL users, see [TiDB User Account Management](#).

Required privileges

- To access TiDB Dashboard when [Security Enhanced Mode \(SEM\)](#) is not enabled on the connected TiDB server, the SQL user should have **all** the following privileges:
 - PROCESS
 - SHOW DATABASES

- CONFIG
 - DASHBOARD_CLIENT
- To access TiDB Dashboard when **Security Enhanced Mode (SEM)** is enabled on the connected TiDB server, the SQL user should have **all** the following privileges:
 - PROCESS
 - SHOW DATABASES
 - CONFIG
 - DASHBOARD_CLIENT
 - RESTRICTED_TABLES_ADMIN
 - RESTRICTED_STATUS_ADMIN
 - RESTRICTED_VARIABLES_ADMIN
 - To modify the configurations on the interface after signing in with TiDB Dashboard, the SQL user must also have the following privilege:
 - SYSTEM_VARIABLES_ADMIN

Note:

Users with high privileges such as **ALL PRIVILEGES** or **SUPER** can sign in with TiDB Dashboard as well. Therefore, to comply with the least privilege principle, it is highly recommended that you create users with the required privileges only to prevent unintended operations. See [Privilege Management](#) for more information on these privileges.

If an SQL user does not meet the preceding privilege requirements, the user fails to sign in with TiDB Dashboard, as shown below.

SQL User Sign In

* Username



Password



Sign in failed: The user does not have sufficient privileges to access TiDB Dashboard. Help

Sign In

Figure 339: insufficient-privileges

Example: Create a least-privileged SQL user to access TiDB Dashboard

- When **Security Enhanced Mode (SEM)** is not enabled on the connected TiDB server, to create an SQL user `dashboardAdmin` that can sign in with TiDB Dashboard, execute the following SQL statements:

```
CREATE USER 'dashboardAdmin'@'%' IDENTIFIED BY '<YOUR_PASSWORD>';
```

```
GRANT PROCESS, CONFIG ON *.* TO 'dashboardAdmin'@'%';
GRANT SHOW DATABASES ON *.* TO 'dashboardAdmin'@'%';
GRANT DASHBOARD_CLIENT ON *.* TO 'dashboardAdmin'@'%';
```

- To modify the configuration items on the interface after signing in with TiDB Dashboard, the user-defined SQL user must be granted with the following privilege:

```
GRANT SYSTEM_VARIABLES_ADMIN ON *.* TO 'dashboardAdmin'@'%';
```

- When [Security Enhanced Mode \(SEM\)](#) is enabled on the connected TiDB server, disable SEM first and execute the following SQL statements to create an SQL user `dashboardAdmin` that can sign in with TiDB Dashboard. After creating the user, enable SEM again:

```
CREATE USER 'dashboardAdmin'@'%' IDENTIFIED BY '<YOUR_PASSWORD>';
GRANT PROCESS, CONFIG ON *.* TO 'dashboardAdmin'@'%';
GRANT SHOW DATABASES ON *.* TO 'dashboardAdmin'@'%';
GRANT DASHBOARD_CLIENT ON *.* TO 'dashboardAdmin'@'%';
GRANT RESTRICTED_STATUS_ADMIN ON *.* TO 'dashboardAdmin'@'%';
GRANT RESTRICTED_TABLES_ADMIN ON *.* TO 'dashboardAdmin'@'%';
GRANT RESTRICTED_VARIABLES_ADMIN ON *.* TO 'dashboardAdmin'@'%';
```

- To modify the configuration items on the interface after signing in with TiDB Dashboard, the user-defined SQL user must be granted with the following privilege:

```
GRANT SYSTEM_VARIABLES_ADMIN ON *.* TO 'dashboardAdmin'@'%';
```

Sign in with TiDB Dashboard

After creating an SQL user that meets the privilege requirements of TiDB Dashboard, you can use this user to [Sign in](#) with TiDB Dashboard.

11.6.1.2.4 Secure TiDB Dashboard

Although you need to sign into TiDB Dashboard before accessing it, TiDB Dashboard is designed to be accessed by trusted user entities by default. When you want to provide TiDB Dashboard to external network users or untrusted users for access, take the following measures to avoid security vulnerabilities.

Enhance security of TiDB users

Set a strong password for the TiDB `root` user

The account system of TiDB Dashboard is consistent with that of TiDB SQL users. By default, TiDB's `root` user has no password, so accessing TiDB Dashboard does not require password authentication, which will give the malicious visitor high privileges, including executing privileged SQL statements.

It is recommended that you set a strong password for the TiDB root user. See [TiDB User Account Management](#) for details. Alternatively, you can disable the TiDB root user.

Create a least-privileged user for TiDB Dashboard

The account system of TiDB Dashboard is consistent with that of TiDB SQL. Users accessing TiDB Dashboard are authenticated and authorized based on TiDB SQL user's privileges. Therefore, TiDB Dashboard requires limited privileges, or merely the read-only privilege. You can configure users to access TiDB Dashboard based on the principle of least privilege, thus avoiding access of high-privileged users.

It is recommended that you create a least-privileged SQL user to access and sign in with TiDB Dashboard. This avoids access of high-privileged users and improves security. See [TiDB Dashboard User Management](#) for details.

Use a firewall to block untrusted access

TiDB Dashboard provides services through the PD client port, which defaults to <http://IP:2379/dashboard/>. Although TiDB Dashboard requires identity authentication, other privileged interfaces (such as <http://IP:2379/pd/api/v1/members>) in PD carried on the PD client port do not require identity authentication and can perform privileged operations. Therefore, exposing the PD client port directly to the external network is extremely risky.

It is recommended that you take the following measures:

- Use a firewall to prohibit a component from accessing **any** client port of the PD component via the external network or untrusted network.

Note:

TiDB, TiKV, and other components need to communicate with the PD component through the PD client port, so do not block access to the internal network between components. Otherwise, the cluster will become unavailable.

- See [Use TiDB Dashboard behind a Reverse Proxy](#) to learn how to configure the reverse proxy to safely provide the TiDB Dashboard service on another port to the external network.

How to open access to TiDB Dashboard port when deploying multiple PD instances

Warning:

This section describes an unsafe access solution, which is for the test environment only. **DO NOT** use this solution in the production environment.

In the test environment, you might need to configure the firewall to open the TiDB Dashboard port for external access.

When multiple PD instances are deployed, only one of the PD instances actually runs TiDB Dashboard, and browser redirection occurs when you access other PD instances. Therefore, you need to ensure that the firewall is configured with the correct IP address. For details of this mechanism, see [Deployment with multiple PD instances](#).

When using the TiUP deployment tool, you can view the address of the PD instance that actually runs TiDB Dashboard by running the following command (replace `CLUSTER_NAME` with the cluster name):

```
tiup cluster display CLUSTER_NAME --dashboard
```

The output is the actual TiDB Dashboard address.

Note:

This feature is available only in the later version of the `tiup cluster` deployment tool (v1.0.3 or later).

Upgrade TiUP Cluster

```
tiup update --self  
tiup update cluster --force
```

The following is a sample output:

```
http://192.168.0.123:2379/dashboard/
```

In this example, the firewall needs to be configured with inbound access for the 2379 port of the 192.168.0.123 open IP, and the TiDB Dashboard is accessed via <http://192.168.0.123:2379/dashboard/>.

Reverse proxy only for TiDB Dashboard

As mentioned in [Use a firewall to block untrusted access](#), the services provided under the PD client port include not only TiDB Dashboard (located at <http://IP:2379/dashboard/>), but also other privileged interfaces in PD (such as <http://IP:2379/pd/api/v1/members>). Therefore, when using a reverse proxy to provide TiDB Dashboard to the external network, ensure that the services **ONLY** with the `/dashboard` prefix are provided (**NOT** all services under the port) to avoid that the external network can access the privileged interface in PD through the reverse proxy.

It is recommended that you see [Use TiDB Dashboard behind a Reverse Proxy](#) to learn a safe and recommended reverse proxy configuration.

Enable TLS for reverse proxy

To further enhance the security of the transport layer, you can enable TLS for reverse proxy, and even introduce mTLS to authenticate user certificates.

See [Configuring HTTPS servers and HAProxy SSL Termination](#) for more details.

Other recommended safety measures

- [Enable TLS Authentication and Encrypt the Stored Data](#)
- [Enable TLS Between TiDB Clients and Servers](#)

11.6.1.3 Access TiDB Dashboard

To access TiDB Dashboard, visit <http://127.0.0.1:2379/dashboard> via your browser. Replace 127.0.0.1:2379 with the actual PD instance address and port.

If multiple PD instances are deployed in your cluster and you can directly access **every** PD instance and port, you can simply replace 127.0.0.1:2379 in the <http://127.0.0.1:2379/dashboard/> address with **any** PD instance address and port.

Note:

If a firewall or reverse proxy is configured and you cannot directly access every PD instance, you might not be able to access TiDB Dashboard. Usually, this is because the firewall or reverse proxy is not correctly configured. See [Use TiDB Dashboard behind Reverse Proxy](#) and [Secure TiDB Dashboard](#) to learn correctly configure the firewall or reverse proxy when multiple PD instances are deployed.

11.6.1.3.1 Browser compatibility

You can use TiDB Dashboard in the following common desktop browsers of a relatively newer version:

- Chrome >= 77
- Firefox >= 68
- Edge >= 17

Note:

If you use the browsers above of earlier versions or other browsers to access TiDB Dashboard, some functions might not work properly.

11.6.1.3.2 Sign in

After accessing TiDB Dashboard, you will be directed to the user login interface, as shown in the image below.

- You can sign in with TiDB Dashboard using the TiDB `root` account. By default, the `root` password is empty.
- If you have created a [User-defined SQL User](#), you can sign in using this account and the corresponding password.



SQL User Sign In

* Username

Password

Sign In

 Switch Language ▾

Figure 340: Login interface

If one of the following situations exists, the login might fail:

- TiDB root user does not exist.
- PD is not started or cannot be accessed.
- TiDB is not started or cannot be accessed.
- Wrong root password.

Once you have signed in, the session remains valid within the next 24 hours. To learn how to sign out, refer to the [Logout](#) section.

11.6.1.3.3 Switch language

The following languages are supported in TiDB Dashboard:

- English
- Chinese (simplified)

In the **SQL User Sign In** page, you can click the **Switch Language** drop-down list to switch the interface language.

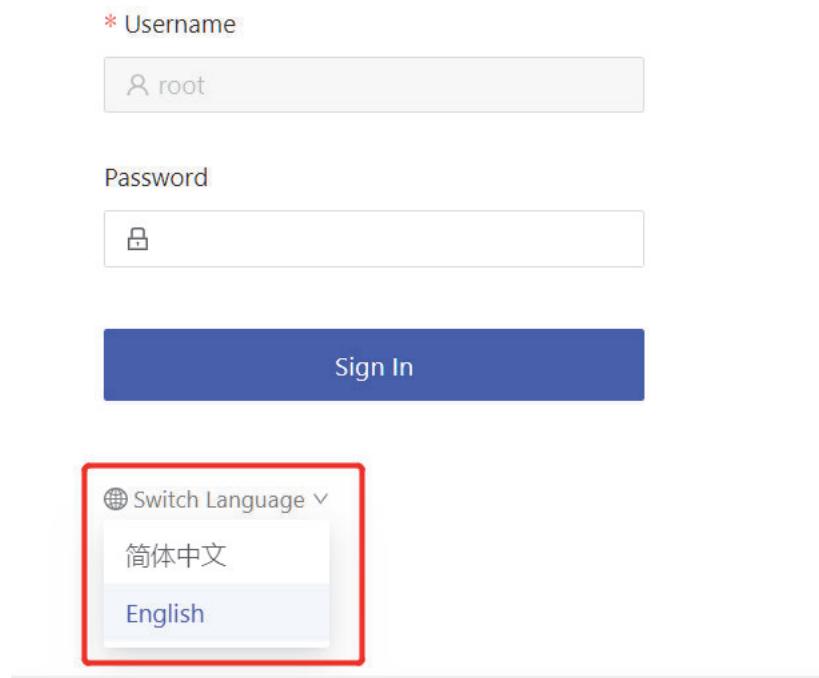


Figure 341: Switch language

11.6.1.3.4 Logout

Once you have logged in, click the login user name in the left navigation bar to switch to the user page. Click the **Logout** button on the user page to log out the current user. After logging out, you need to re-enter your username and password.

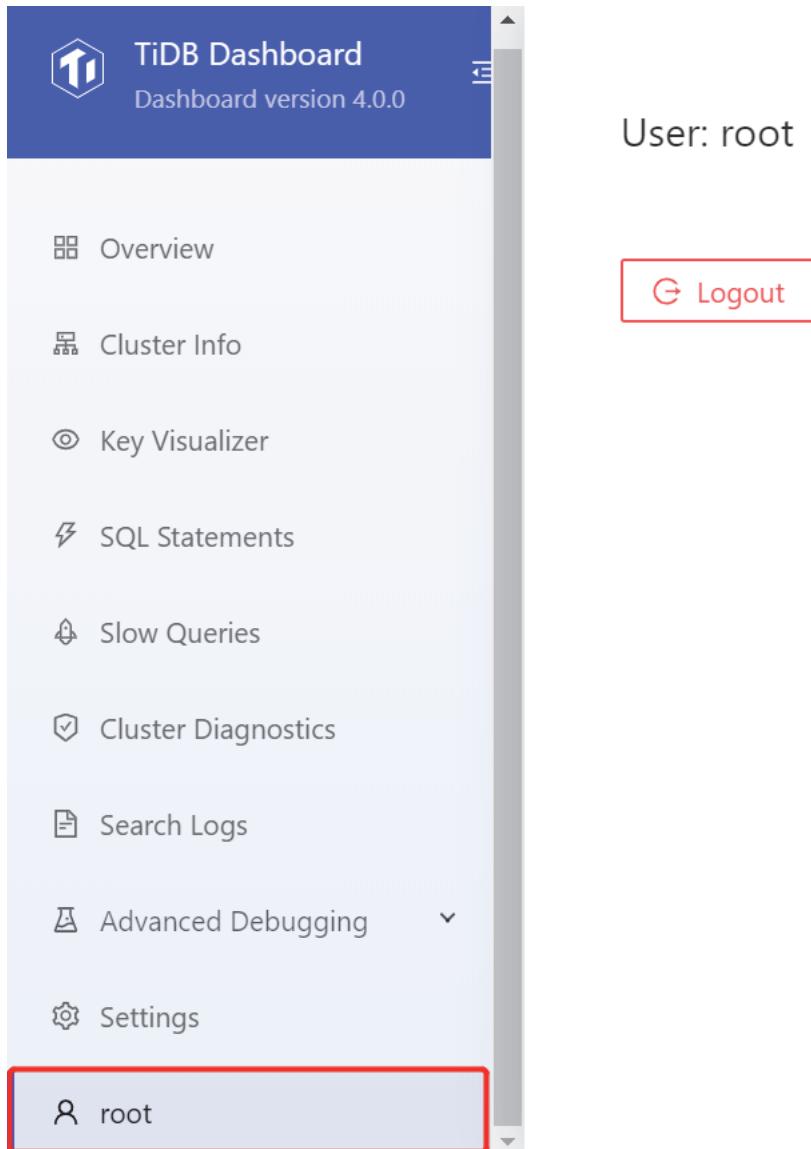


Figure 342: Logout

11.6.1.4 Overview Page

This page shows the overview of the entire TiDB cluster, including the following information:

- Queries per second (QPS) of the entire cluster.
- The query latency of the entire cluster.
- The SQL statements that have accumulated the longest execution time over the recent period.
- The slow queries whose execution time over the recent period exceeds the threshold.
- The node count and status of each instance.
- Monitor and alert messages.

11.6.1.4.1 Access the page

After logging into TiDB Dashboard, the overview page is entered by default, or you can click **Overview** on the left navigation menu to enter this page:

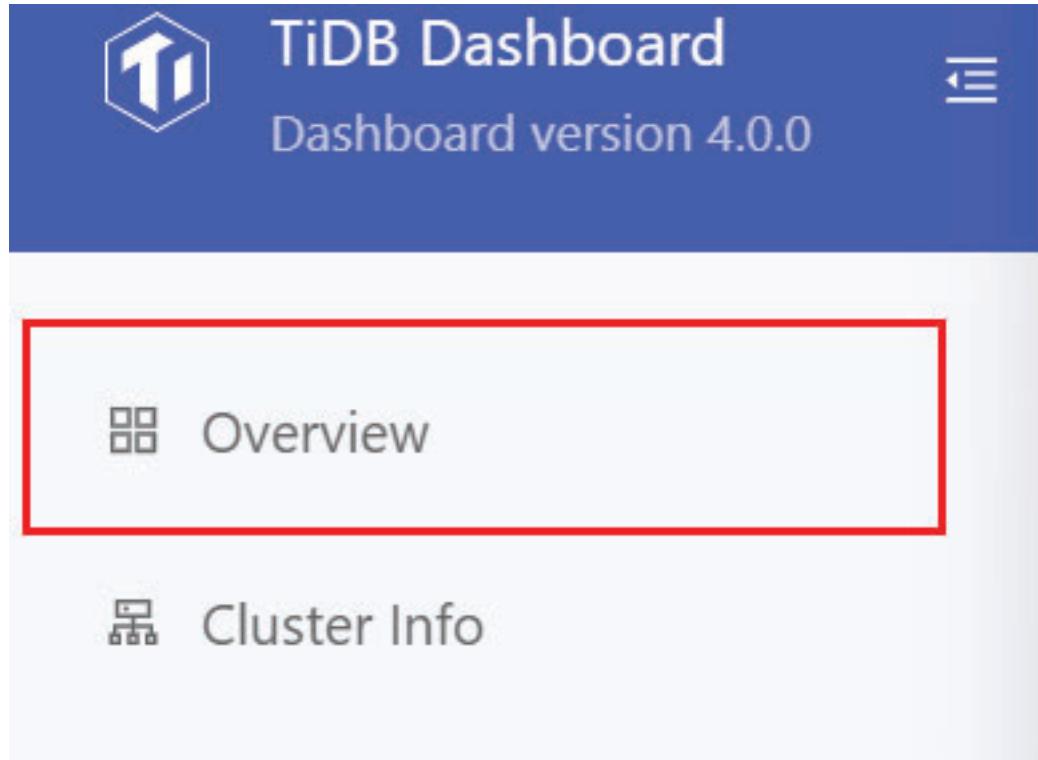


Figure 343: Enter overview page

11.6.1.4.2 QPS

This area shows the number of successful and failed queries per second for the entire cluster over the recent hour:

QPS

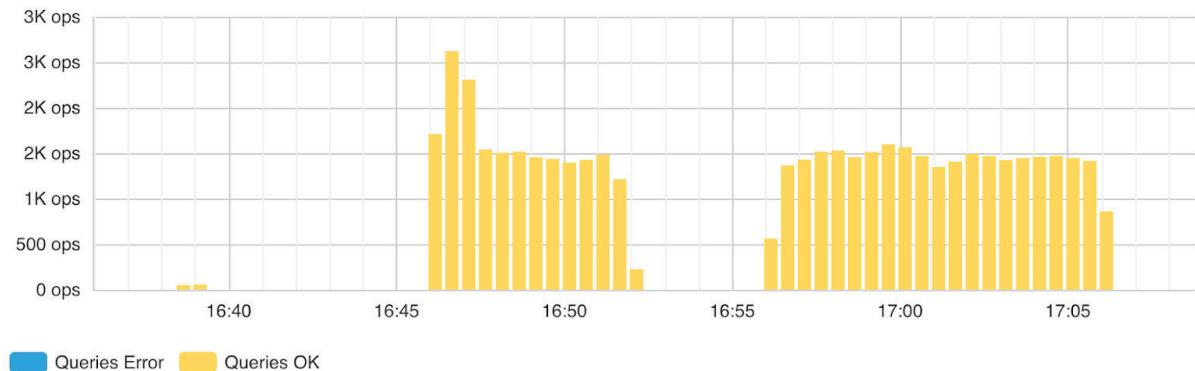


Figure 344: QPS

Note:

This feature is available only in the cluster where the Prometheus monitoring component is deployed. If Prometheus is not deployed, an error will be displayed.

11.6.1.4.3 Latency

This area shows the latency of 99.9%, 99%, and 90% of queries in the entire cluster over the recent one hour:

Latency

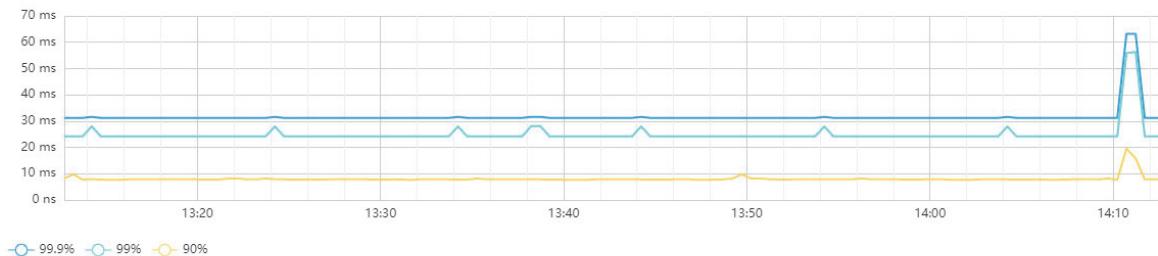


Figure 345: Latency

Note:

This feature is available only on the cluster where the Prometheus monitoring component is deployed. If Prometheus is not deployed, an error will be displayed.

11.6.1.4.4 Top SQL statements

This area shows the ten types of SQL statements that have accumulated the longest execution time in the entire cluster over the recent period. SQL statements with different query parameters but of the same structure are classified into the same SQL type and displayed in the same row:

[Top SQL Statements >](#) Today at 10:00 AM ~ Today at 10:30 AM

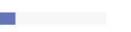
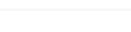
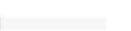
SQL Template ⓘ	Total Latency ⓘ ↓	Mean Latency ⓘ	Database ⓘ
SELECT count (k) FROM sbtest1 WHERE...	9.8 s		3.5 ms
CREATE INDEX k_1 ON sbtest1 (k)	4.0 s		4.0 s
SELECT * FROM information_schema.cl...	1.5 s		1.5 s
INSERT INTO sbtest1 (k, c, pad) VAL...	823.4 ms		45.7 ms
CREATE TABLE sbtest1 (id integer N...	86.2 ms		86.2 ms
SELECT @ @global.tidb_enable_stmt_s...	54.0 ms		10.8 ms
SELECT DISTINCT stmt_type FROM info...	46.7 ms		9.3 ms
SELECT DISTINCT floor (unix_timesta...	43.2 ms		8.6 ms
SELECT *, unix_timestamp (time) AS ...	29.6 ms		7.4 ms
INSERT INTO sbtest1 (k, c, pad) VAL...	22.7 ms		22.7 ms

Figure 346: Top SQL

The information shown in this area is consistent with the more detailed [SQL Statements Page](#). You can click the **Top SQL Statements** heading to view the complete list. For details of the columns in this table, see [SQL Statements Page](#).

Note:

This feature is available only on the cluster where SQL Statements feature is enabled.

11.6.1.4.5 Recent slow queries

By default, this area shows the latest 10 slow queries in the entire cluster over the recent 30 minutes:

[Recent Slow Queries >](#) Today at 3:49 PM ~ Today at 4:19 PM

SQL ⓘ	Finish Time ⓘ ↓	Latency ⓘ	Max Memory ⓘ
ANALYZE TABLE `tpcc`.`bmsql_order`;	Today at 4:18 PM	1.1 s	0 B
ANALYZE TABLE `tpcc`.`bmsql_new_order`;	Today at 4:17 PM	406.5 ms	0 B
ANALYZE TABLE `tpcc`.`bmsql_customer`;	Today at 4:17 PM	3.4 s	0 B
ANALYZE TABLE `tpcc`.`bmsql_new_order`;	Today at 4:15 PM	394.5 ms	0 B
ANALYZE TABLE `tpcc`.`bmsql_stock`;	Today at 4:15 PM	3.5 s	0 B
ANALYZE TABLE `tpcc`.`bmsql_new_order`;	Today at 4:13 PM	414.6 ms	0 B
ANALYZE TABLE `tpcc`.`bmsql_customer`;	Today at 4:09 PM	2.6 s	0 B
ANALYZE TABLE `tpcc`.`bmsql_order_line`;	Today at 4:08 PM	1.6 s	0 B
ANALYZE TABLE `tpcc`.`bmsql_new_order`;	Today at 4:07 PM	406.9 ms	0 B
ANALYZE TABLE `tpcc`.`bmsql_order`;	Today at 4:07 PM	971.0 ms	0 B

Figure 347: Recent slow queries

By default, the SQL query that is executed longer than 300 milliseconds is counted as a slow query and displayed on the table. You can change this threshold by modifying the `tidb_slow_log_threshold` variable or the `slow-threshold` TiDB parameter.

The content displayed in this area is consistent with the more detailed [Slow Queries Page](#). You can click the **Recent Slow Queries** title to view the complete list. For details of the columns in this table, see this [Slow Queries Page](#).

Note:

This feature is available only in the cluster with slow query logs enabled. By default, slow query logs are enabled in the cluster deployed using TiUP.

11.6.1.4.6 Instances

This area summarizes the total number of instances and abnormal instances of TiDB, TiKV, PD, and TiFlash in the entire cluster:

Instances >

TiDB: 1 Up / 0 Down

TiKV: 0 Up / 1 Down

PD: 1 Up / 0 Down

Figure 348: Instances

The statuses in the image above are described as follows:

- Up: The instance is running properly (including the offline storage instance).

- Down: The instance is running abnormally, such as network disconnection, process crash, and so on.

Click the **Instance** title to enter the [Cluster Info Page](#) that shows the detailed running status of each instance.

11.6.1.4.7 Monitor and alert

This area provides links for you to view detailed monitor and alert:

Monitor & Alert

[View Metrics >](#)

[View 1 Alerts >](#)

[Run Diagnostics >](#)

Figure 349: Monitor and alert

- **View Metrics:** Click this link to jump to the Grafana dashboard where you can view detailed monitoring information of the cluster. For details of each monitoring metric in the Grafana dashboard, see [monitoring metrics](#).
- **View Alerts:** Click this link to jump to the AlertManager page where you can view detailed alert information of the cluster. If alerts exist in the cluster, the number of alerts is directly shown in the link text.
- **Run Diagnostics:** Click this link to jump to the more detailed [cluster diagnostics page](#).

Note:

The **View Metrics** link is available only in the cluster where the Grafana node is deployed. The **View Alerts** link is available only in the cluster where the AlertManager node is deployed.

11.6.1.5 TiDB Dashboard Cluster Information Page

On the cluster information page, you can view the running status of TiDB, TiKV, PD, TiFlash components in the entire cluster and the running status of the host on which these components are located.

11.6.1.5.1 Access the page

You can use one of the following two methods to access the cluster information page:

- After logging into TiDB Dashboard, click **Cluster Info** on the left navigation menu:

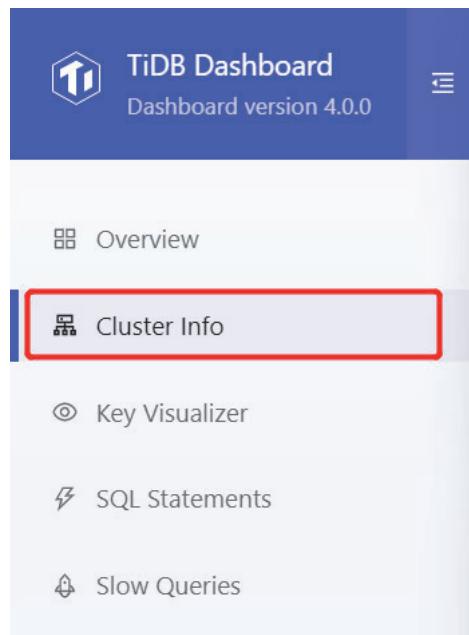
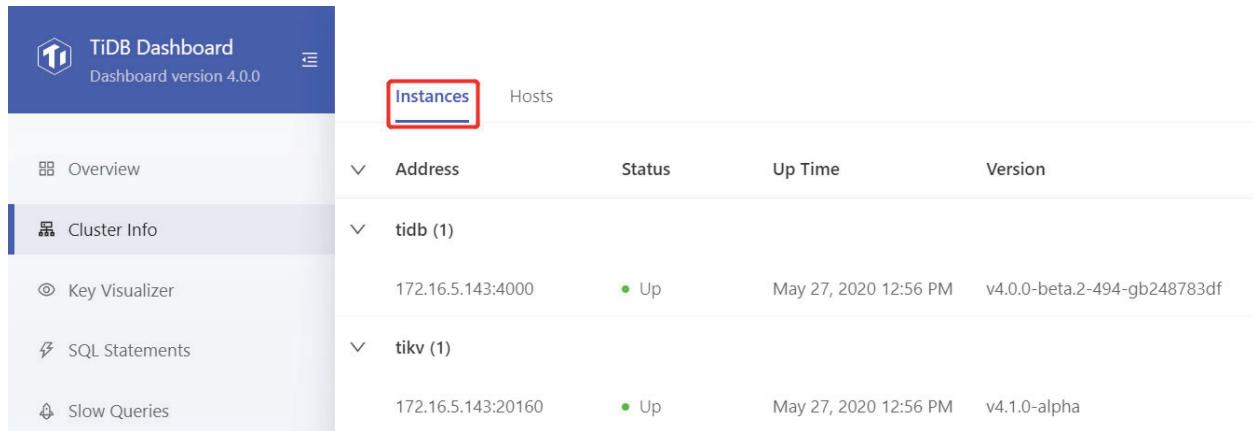


Figure 350: Access cluster information page

- Visit http://127.0.0.1:2379/dashboard/#/cluster_info/instance in your browser. Replace 127.0.0.1:2379 with the actual PD instance address and port.

11.6.1.5.2 Instance list

Click **Instances** to view the list of instances:



	Address	Status	Up Time	Version
tidb (1)	172.16.5.143:4000	Up	May 27, 2020 12:56 PM	v4.0.0-beta.2-494-gb248783df
tikv (1)	172.16.5.143:20160	Up	May 27, 2020 12:56 PM	v4.1.0-alpha

Figure 351: Instance list

This instance list shows the overview information of all instances of TiDB, TiKV, PD, and TiFlash components in the cluster.

The list includes the following information:

- Address: The instance address.
- Status: The running status of the instance.
- Up Time: The start time of the instance.
- Version: The instance version number.
- Deployment directory: The directory in which the instance binary file is located.
- Git Hash: The Git Hash value corresponding to the instance binary file.

An instance has the following running status:

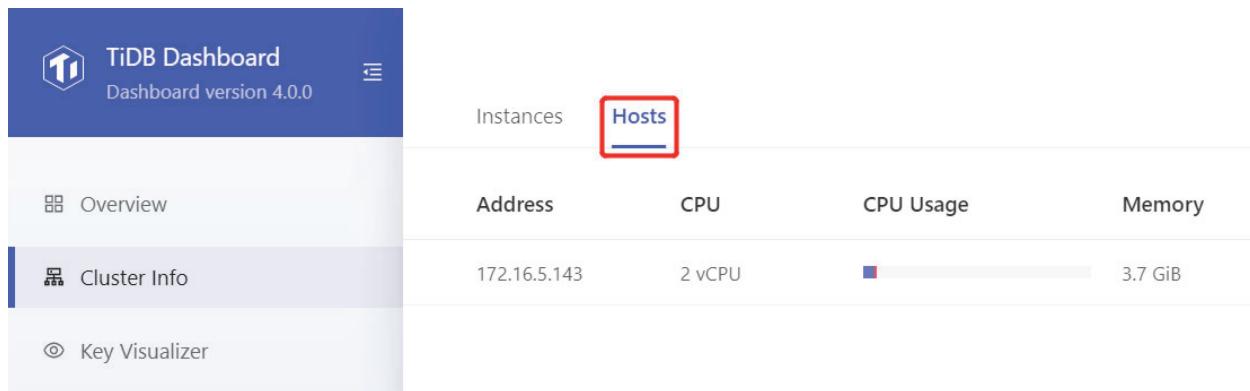
- Up: The instance is running properly.
- Down or Unreachable: The instance is not started or a network problem exists on the corresponding host.
- Tombstone: The data on the instance has been completely migrated out and the scaling-in is complete. This status exists only on TiKV or TiFlash instances.
- Leaving: The data on the instance is being migrated out and the scaling-in is in process. This status exists only on TiKV or TiFlash instances.
- Unknown: The running state of the instance is unknown.

Note:

Some columns in the table can be displayed only when the instance is up.

11.6.1.5.3 Host list

Click **Hosts** to view the list of hosts:



The screenshot shows the TiDB Dashboard interface with the title "TiDB Dashboard" and "Dashboard version 4.0.0". On the left, there's a sidebar with three items: "Overview", "Cluster Info", and "Key Visualizer". The "Cluster Info" item is currently selected. At the top right, there are tabs for "Instances" and "Hosts", with "Hosts" being the active tab and highlighted with a red box. Below the tabs is a table with four columns: "Address", "CPU", "CPU Usage", and "Memory". A single host entry is shown: Address 172.16.5.143, CPU 2 vCPU, CPU Usage (represented by a progress bar), and Memory 3.7 GiB.

Figure 352: Host list

This host list shows the running status of hosts that correspond to all instances of TiDB, TiKV, PD, and TiFlash components in the cluster.

The list includes the following information:

- Address: The Host IP address.
- CPU: The number of logical cores of the host CPU.
- CPU Usage: The user-mode and kernel-mode CPU usage in the current 1 second.
- Memory: The total physical memory size of the host.
- Memory Usage: The current memory usage of the host.
- Disk: The file system of the disk on the host on which the instance is running and the mounting path of this disk.
- Disk Usage: The space usage of the disk on the host on which the instance is running.

Note:

The host list information is provided by each instance process, so when all instances on the host are down, the host information is not displayed.

11.6.1.6 Key Visualizer Page

The Key Visualizer page of TiDB Dashboard is used to analyze the usage of TiDB and troubleshoot traffic hotspots. This page visually shows the traffic of the TiDB cluster over a period of time.

11.6.1.6.1 Access Key Visualizer page

You can use one of the following two methods to access the Key Visualizer page:

- After logging into TiDB Dashboard, click **Key Visualizer** on the left navigation menu:

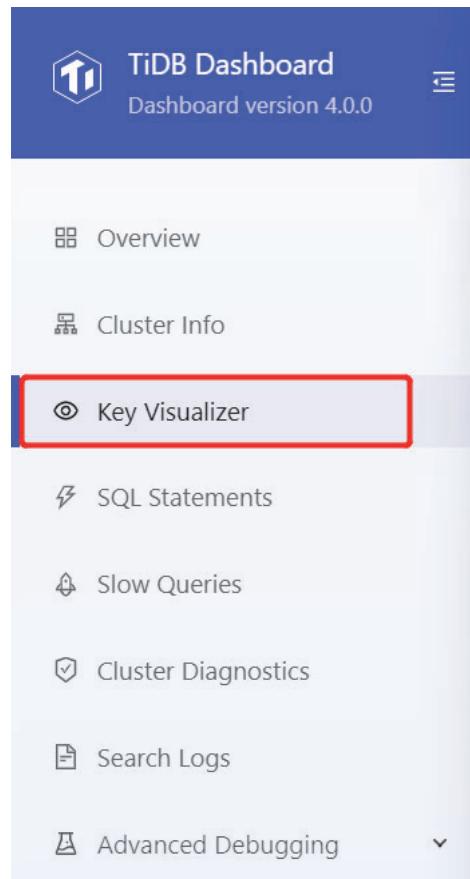


Figure 353: Access Key Visualizer

- Visit <http://127.0.0.1:2379/dashboard/#/keyviz> in your browser. Replace 127.0.0.1:2379 ↳ with the actual PD instance address and port.

11.6.1.6.2 Interface demonstration

The following image is a demonstration of the Key Visualizer page:

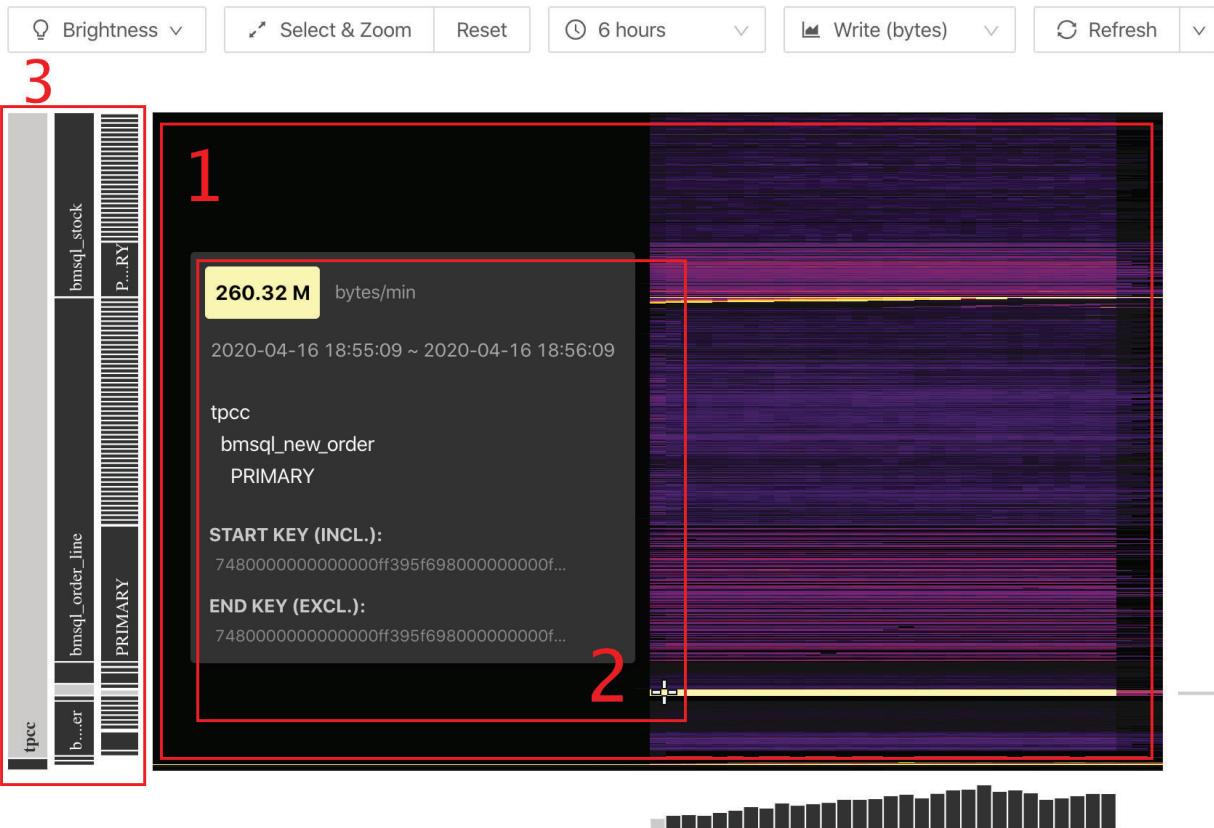


Figure 354: Key Visualizer page

From the interface above, you can see the following objects:

- A large heatmap that shows changes of the overall traffic over time.
- The detailed information of a certain coordinate point.
- Information of tables, indexes, and so on (on the left side of the heatmap).

11.6.1.6.3 Basic concepts

This section introduces the basic concepts that relate to Key Visualizer.

Region

In a TiDB cluster, the stored data is distributed among TiKV instances. Logically, TiKV is a huge and orderly key-value map. The whole key-value space is divided into many segments and each segment consists of a series of adjacent keys. Such segment is called a **Region**.

For detailed introduction of Region, refer to [TiDB Internal \(I\) - Data Storage](#).

Hotspot

When you use the TiDB database, the hotspot issue is typical, where high traffic is concentrated on a small range of data. Because consecutive data ranges are often processed on the same TiKV instance, the TiKV instance on which the hotspot occurs becomes the performance bottleneck of the whole application. The hotspot issue often occurs in the following scenarios:

- Write adjacent data into a table with the `AUTO_INCREMENT` primary key, which causes a hotspot issue on this table.
- Write adjacent time data into the time index of a table, which causes a hotspot issue on the table index.

For more details about hotspot, refer to [Highly Concurrent Write Best Practices](#)

Heatmap

The heatmap is the core part of Key Visualizer, which shows the change of a metric over time. The X-axis of the heatmap indicates the time. The Y-axis of the heatmap indicates the consecutive Regions based on key ranges that cover all schemas and tables of the TiDB cluster.

Colder colors in the heatmap indicate lower read and write traffic of Regions in that period of time. Hotter (brighter) colors indicate higher traffic.

Region compression

A TiDB cluster might have up to hundreds of thousands of Regions. It is difficult to display so many Regions on screen. Therefore, on each heatmap, these Regions are compressed into 1,500 consecutive ranges, each range called a Bucket. In the heatmap, because hotter instances need more attention, Key Visualizer tends to compress a large number of Regions with lower traffic into one Bucket, and displays the Region with higher traffic also in one Bucket.

11.6.1.6.4 Use Key Visualizer

This section introduces how to use Key Visualizer.

Settings

To use the Key Visualizer page for the first time, you need to manually enable this feature on the **Settings** page. Follow the page guide and click **Open Settings** to open the settings page:



Feature Not Enabled

Key Visualizer feature is not enabled so that visual reports cannot be viewed. You can modify settings to enable the feature and wait for new data being collected.

[Open Settings](#)

Figure 355: Feature disabled

After this feature is enabled, you can open the settings page by clicking the **Settings** icon in the upper right corner:

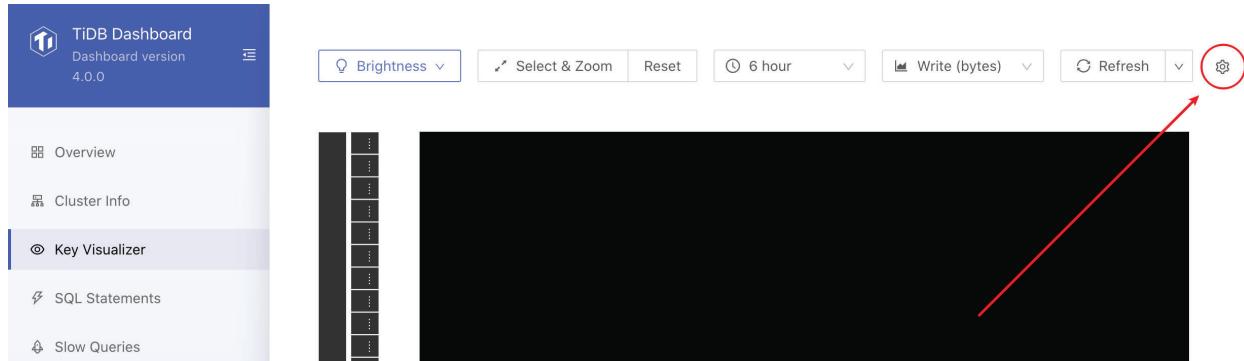


Figure 356: Settings icon

The settings page is shown as follows:

Settings

X

Enable



Save

Cancel

Figure 357: Settings page

Set whether to start data collection through the switch, and click **Save** to take effect. After enabling the feature, you can see that the toolbar is available:

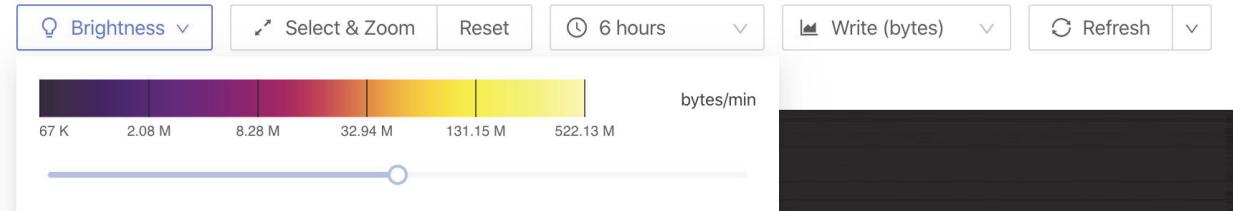


Figure 358: Toolbar

After this feature is enabled, data collection is going on at the backend. You can see the heatmap shortly.

Observe a certain period of time or Region range

When you open Key Visualizer, the heatmap of the entire database over the recent six hours is displayed by default. In this heatmap, the closer to the right side (current time), the shorter the time interval corresponding to each column of Buckets. If you want to observe a specific time period or a specific Region range, you can zoom in to get more details. The specific instructions are as follows:

1. Scroll up or down in the heatmap.
2. Click and drag one of the following buttons to select the range.

- Click the **Select & Zoom** button. Then click and drag this button to select the area to zoom in.

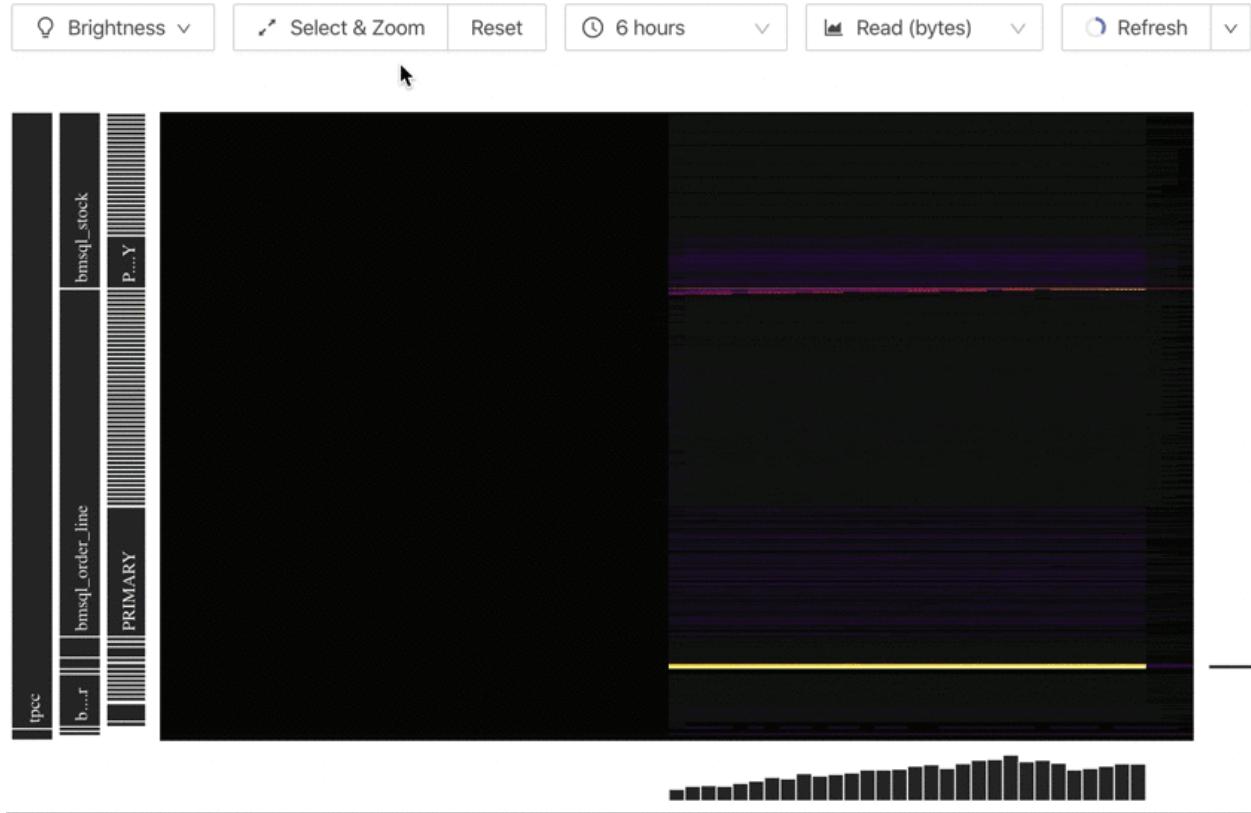


Figure 359: Selection box

- Click the **Reset** button to reset the Region range to the entire database.
- Click the **time selection box** (at the position of 6 hours on the interface above) and select the observation time period again.

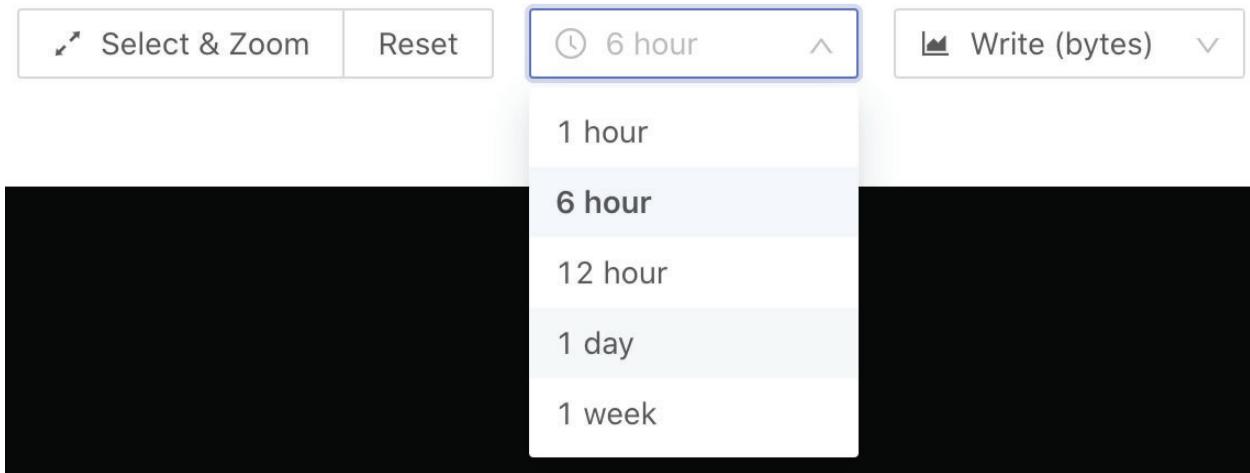


Figure 360: Select time

Note:

If you follow step 2 in the instruction above, the heatmap is redrawn, which might be very different from the original heatmap. This difference is normal because if you observe in more detail, the granularity of the Region compression has changed, or the basis of hot in a specific range has changed.

Adjust brightness

The heatmap uses colors of different brightnesses to indicate the Bucket traffic. Colder colors in the heatmap indicate lower read and write traffic of the Region in that period of time. Hotter (brighter) colors indicate higher traffic. If the color is too cold or too hot, it is difficult to observe in details. In this situation, you can click the **Brightness** button and then use the slider to adjust the brightness of the page.

Note:

When Key Visualizer displays the heatmap of an area, it defines the basis of being cold and hot according to the traffic of this area. When the traffic in the entire area is relatively even, even if the overall traffic is low in value, you might still see a large bright-colored area. Remember to include the value into your analysis.

Select metrics

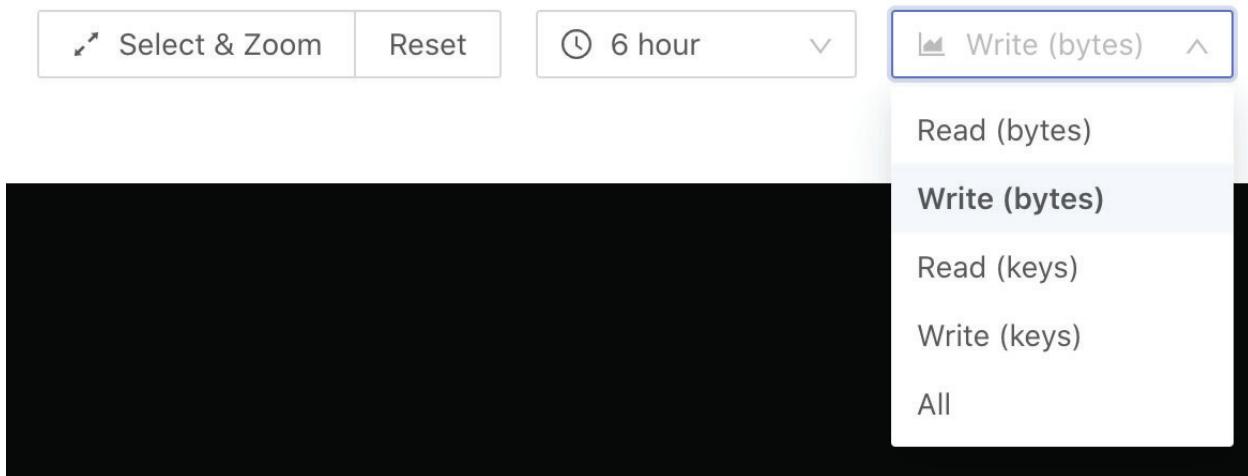


Figure 361: Select metrics

You can view a metric you are interested in by selecting this metric in the **metrics selection box** (at the Write (bytes) position in the interface above):

- Read (bytes): Read traffic.
- Write (bytes): Write traffic.
- Read (keys): The number of read rows.
- Write (keys): The number of written rows.
- All: The sum of read and write traffic.

Refresh and automatic refresh

To regain a heatmap based on the current time, click the **Refresh** button. If you need to observe the traffic distribution of the database in real time, click the down arrow on the right side of the **Refresh** button and select a fixed time interval for the heatmap to automatically refresh at this interval.

Note:

If you adjust the time range or Region range, the automatic refresh is disabled.

See Bucket details

You can hover your mouse over the Bucket you are interested in to view the detailed information of this Region range. The image below is an example of this information:

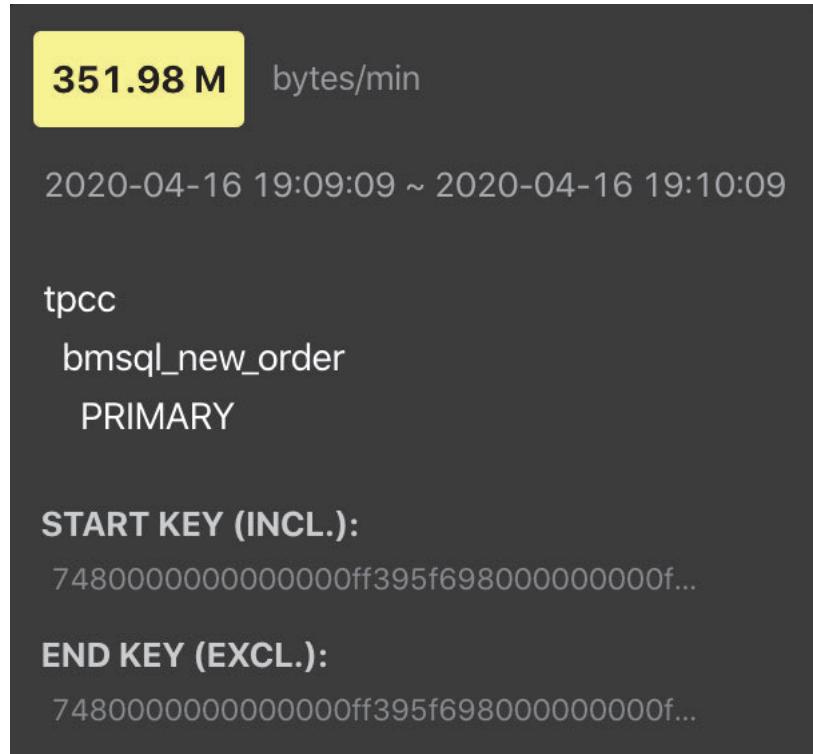


Figure 362: Bucket details

If you want to copy this information, click a Bucket. Then, the page with relevant details is temporarily pinned. Click on the information, and you have copied it to the clipboard.

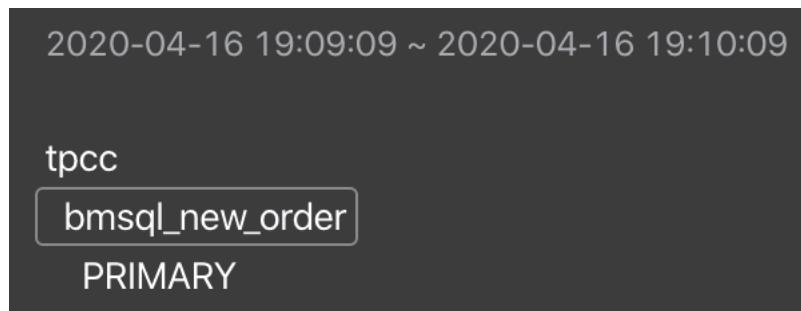


Figure 363: Copy Bucket details

11.6.1.6.5 Common heatmap types

This section shows and interprets four common types of heatmap in Key Visualizer.
Evenly distributed workload

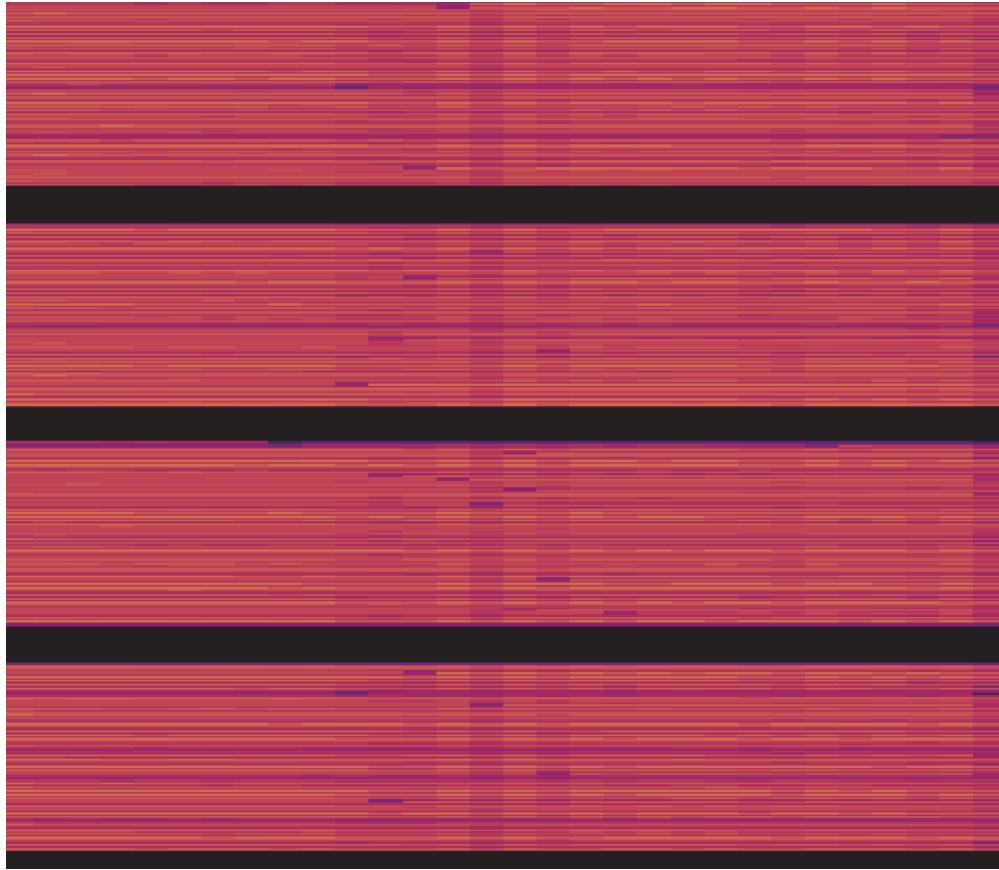


Figure 364: Balanced

In the heatmap above, bright and dark colors are a fine-grained mix. This indicates that reads or writes are evenly distributed over time and among key ranges. The workload is evenly distributed to all nodes, which is ideal for a distributed database.

Periodically reads and writes

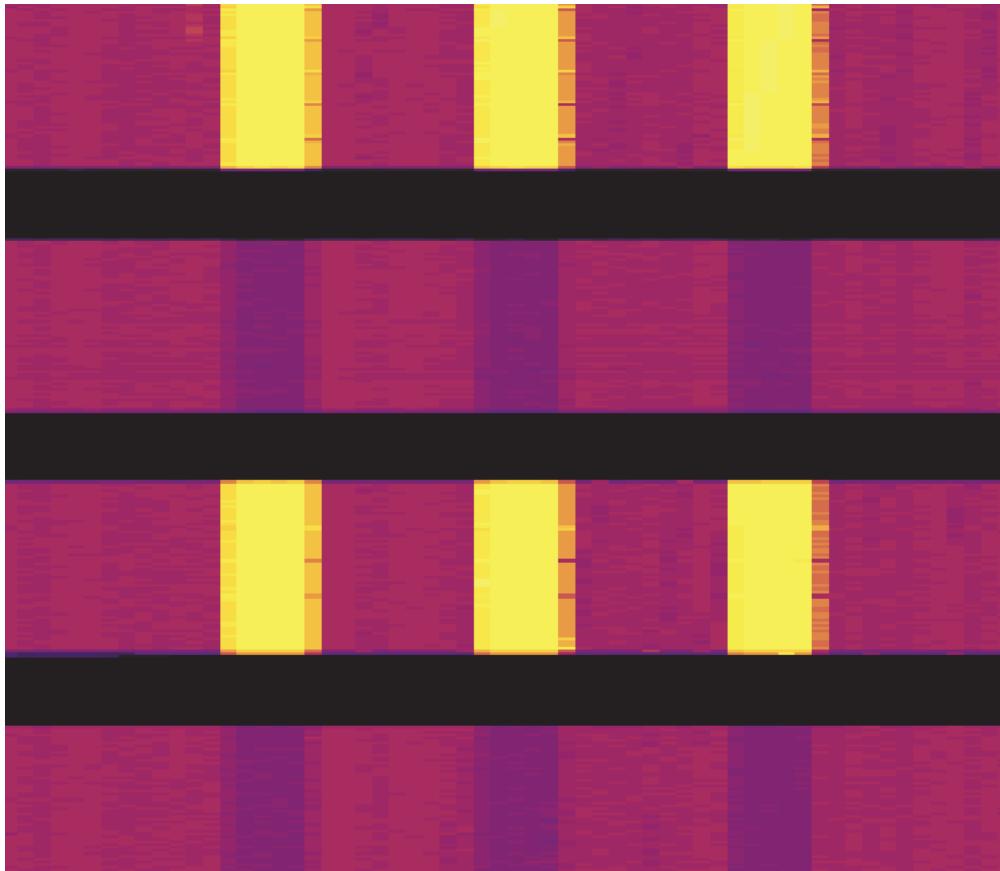


Figure 365: Periodically

In the heatmap above, there is an alternating brightness and darkness along the X-axis (time) but the brightness is relatively even along the Y-axis (Region). This indicates that the reads and writes change periodically, which might occur in scenarios of periodically scheduled tasks. For example, the big data platform periodically extracts data from TiDB every day. In this kind of scenarios, pay attention to whether the resources are sufficient during peak usage.

Concentrated reads or writes

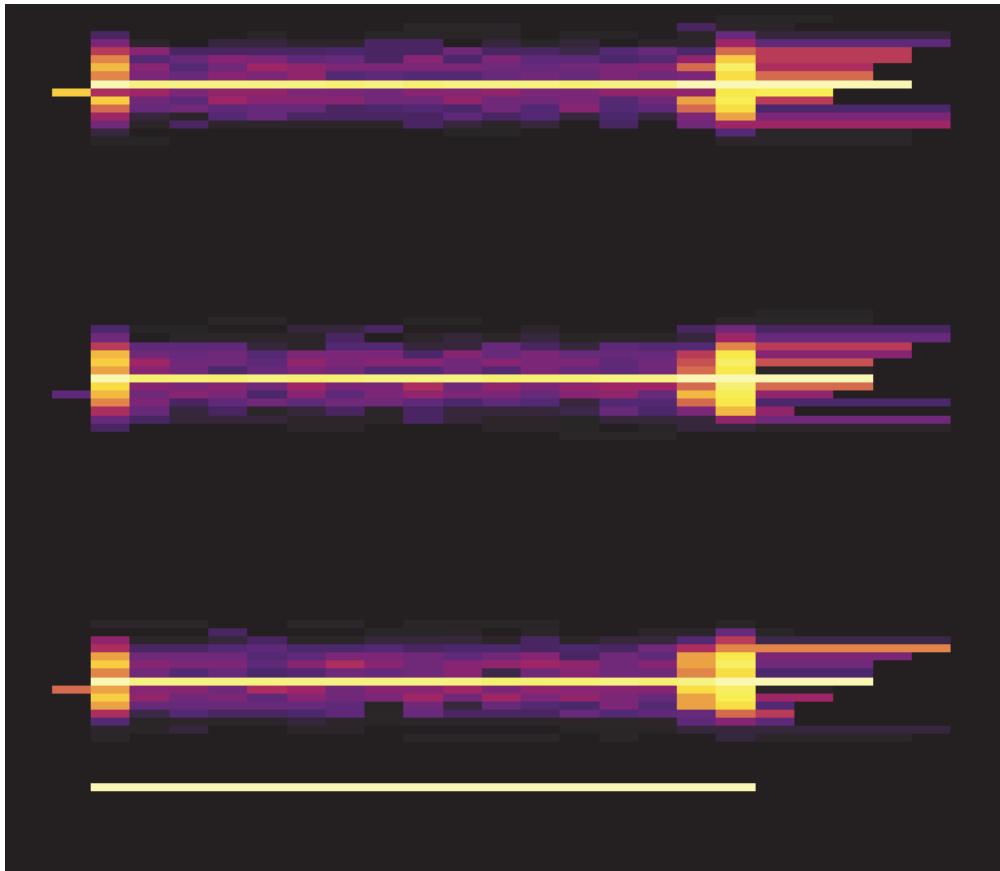


Figure 366: Concentrated

In the heatmap above, you can see several bright lines. Along the Y-axis, the fringes around the bright lines are dark, which indicates that the Regions corresponding to bright lines have high read and write traffic. You can observe whether the traffic distribution is expected by your application. For example, when all services are associated with the user table, the overall traffic of the user table can be high, so it is reasonable to show bright lines in the heatmap.

In addition, the height of the bright lines (the thickness along the Y-axis) is important. Because TiKV has its own Region-based hotspot balancing mechanism, the more Regions involved in the hotspot, the better it is for balancing traffic across all TiKV instances. The thicker and more bright lines indicate that hotspots are more scattered, and TiKV is better used. The thinner and fewer bright lines indicate that hotspots are more concentrated, and the hotspot issue is more obvious in TiKV, which might require manual intervention.

Sequential reads or writes

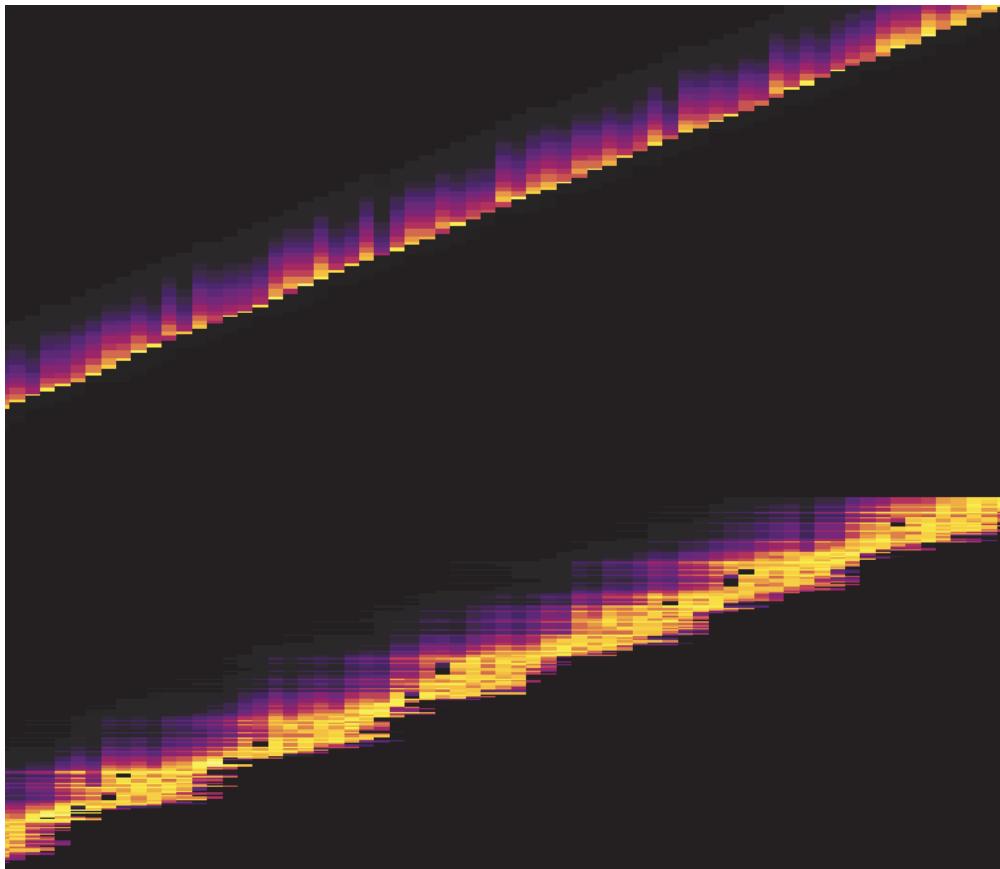


Figure 367: Sequential

In the heatmap above, you can see a bright line. This means that the data reads or writes are sequential. Typical scenarios of sequential data reads or writes are importing data or scanning tables and indexes. For example, you continuously write data to tables with auto-increment IDs.

Regions in the bright areas are the hotspots of read and write traffic, which often become the performance bottleneck of the entire cluster. In this situation, you might need to readjust the primary key for the application. By doing this, you scatter Regions much as possible to spread the pressure across multiple Regions. You can also schedule application tasks during the low-peak period.

Note:

In this section, only the common types of heatmap are shown. Key Visualizer actually displays the heatmap of all schemas and tables in the entire cluster, so you might see different types of heatmap in different areas, or mixed results of multiple heatmap types. Use the heatmap based on the actual situation.

11.6.1.6.6 Address hotspot issues

TiDB has some built-in features to mitigate the common hotspot issue. Refer to [Highly Concurrent Write Best Practices](#) for details.

11.6.1.7 TiDB Dashboard Metrics Relation Graph

TiDB Dashboard metrics relation graph is a feature introduced in v4.0.7. This feature presents a relation graph of the monitoring data of each internal process's duration in a TiDB cluster. The aim is to help you quickly understand the duration of each process and their relations.

11.6.1.7.1 Access graph

Log into TiDB Dashboard, click **Cluster Diagnostics** on the left navigation menu, and you can see the page of generating the metrics relation graph.

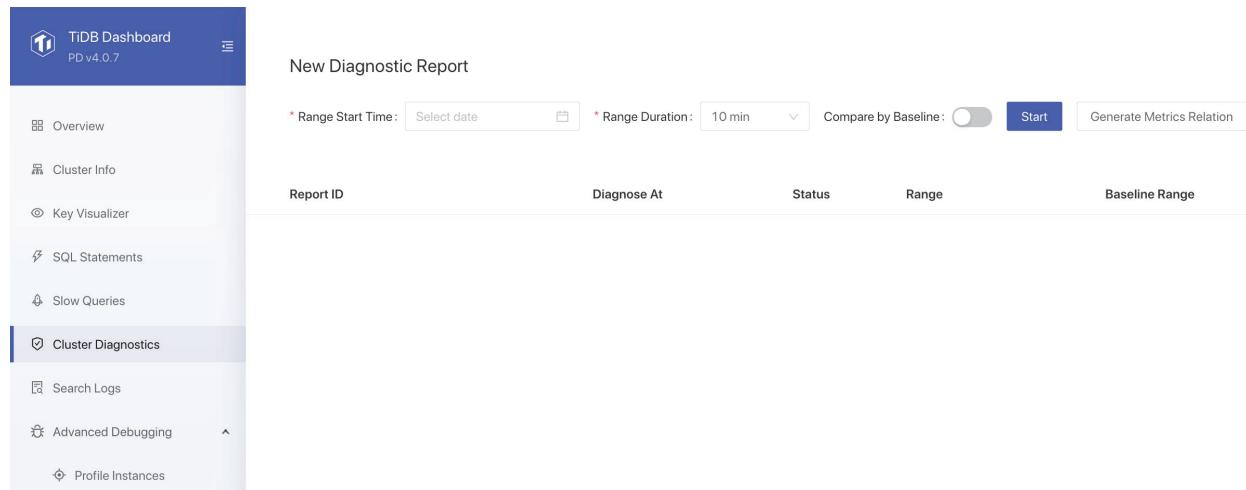
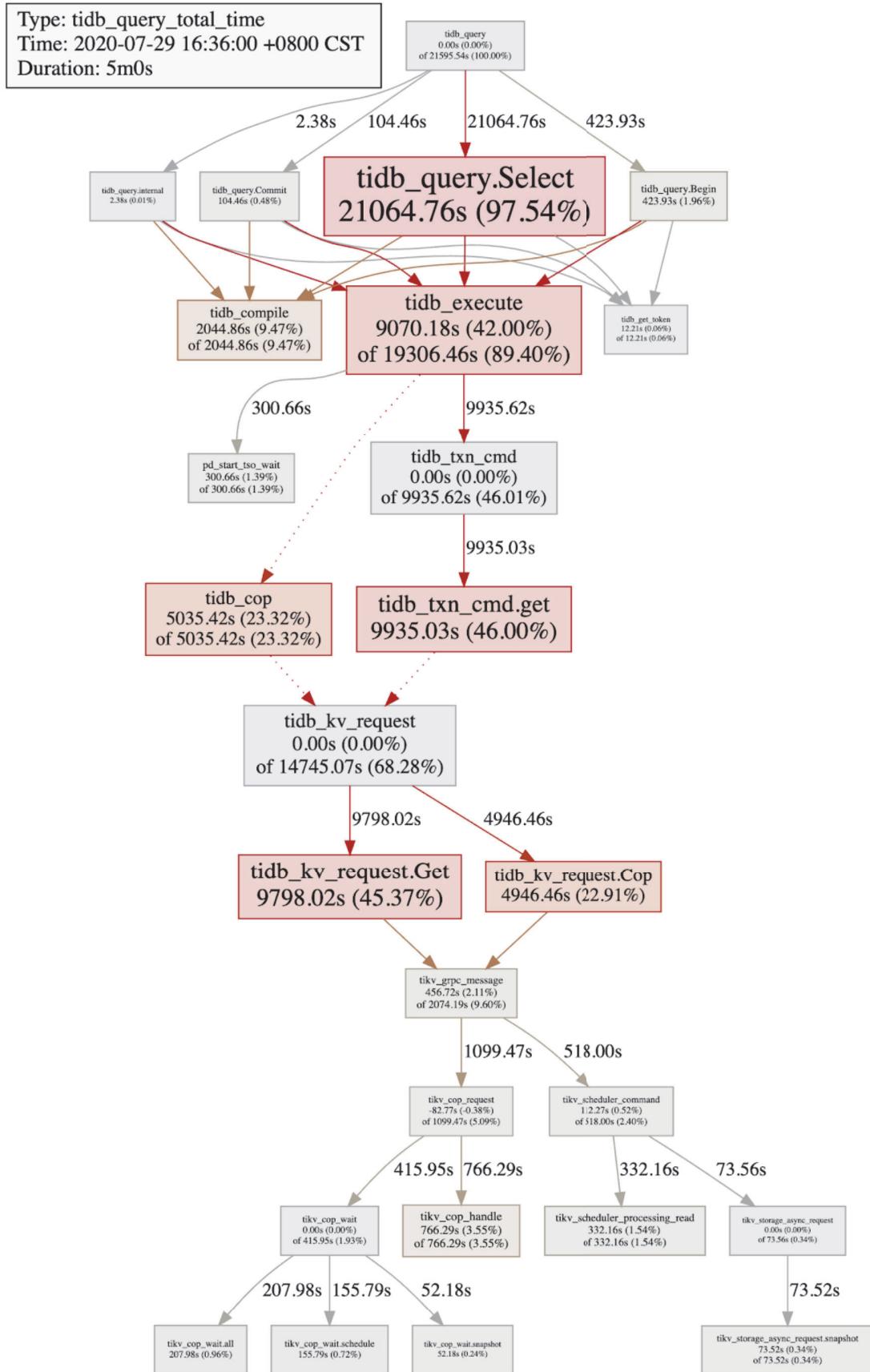


Figure 368: Metrics relation graph homepage

After setting **Range Start Time** and **Range Duration**, click the **Generate Metrics Relation** button and you will enter the page of metrics relation graph.

11.6.1.7.2 Understand graph

The following image is an example of the metrics relation graph. This graph illustrates the proportion of each monitoring metric's duration to the total query duration in a TiDB cluster within 5 minutes after 2020-07-29 16:36:00. The graph also illustrates the relations of each monitoring metric.


 Figure 369: Metrics relation graph example
 2143

For example, the node meaning of the `tidb_execute` monitoring metric is as follows:

- The total duration of the `tidb_execute` monitoring metric is 19306.46 seconds, which accounts for 89.4% of the total query duration.
- The duration of the `tidb_execute` node itself is 9070.18 seconds, which accounts for 42% of the total query duration.
- Hover your mouse over the box area, and you can see the detailed information of the metric, including the total duration, the average duration, and the average P99 (99th percentile) duration.

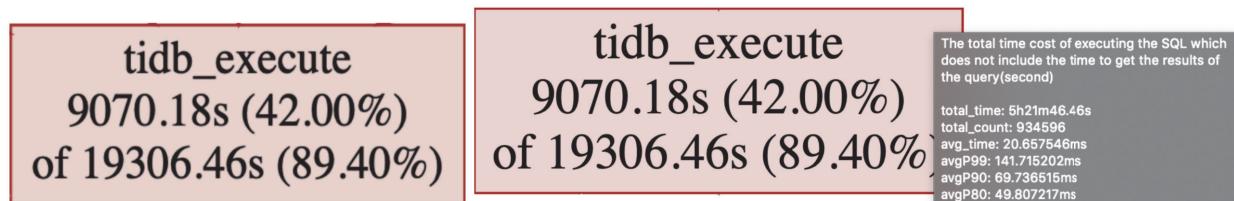


Figure 370: `tidb_execute` node example

Node information

Each box area represents a monitoring metric and provides the following information:

- The name of the monitoring metric
- The total duration of the monitoring metric
- The proportion of the metric's total duration to the total query duration

The total duration of the metric node = the duration of the metric node itself + the duration of its child nodes. Therefore, the metric graph of some nodes displays the proportion of the node itself's duration to the total duration, such as the graph of `tidb_execute`.

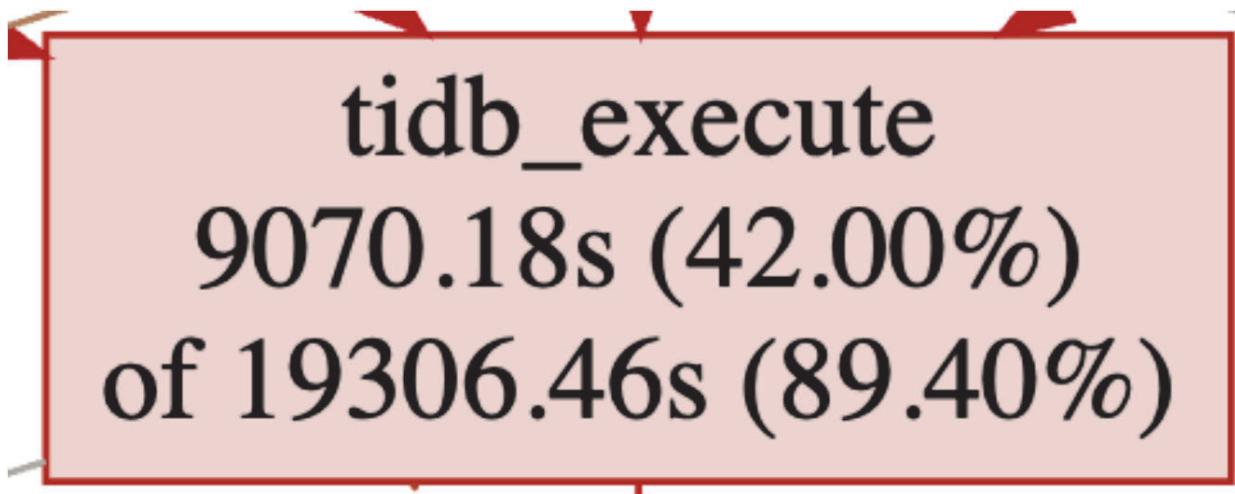


Figure 371: tidb_execute node example1

- `tidb_execute` is the name of the monitoring metric, which represents the execution duration of a SQL query in the TiDB execution engine.
- `19306.46s` represents that total duration of the `tidb_execute` metric is 19306.46 seconds. 89.40% represents that 19306.46 seconds account for 89.40% of the total time consumed for all SQL queries (including user SQL queries and TiDB's internal SQL queries). The total query duration is the total duration of `tidb_query`.
- `9070.18s` represents that the total execution duration of the `tidb_execute` node itself is 9070.18 seconds, and the rest is the time consumed by its child nodes. 42.00% represents that 9070.18 seconds account for 42.00% of the total query duration of all queries.

Hover your mouse over the box area and you can see more details of the `tidb_execute` metric node:

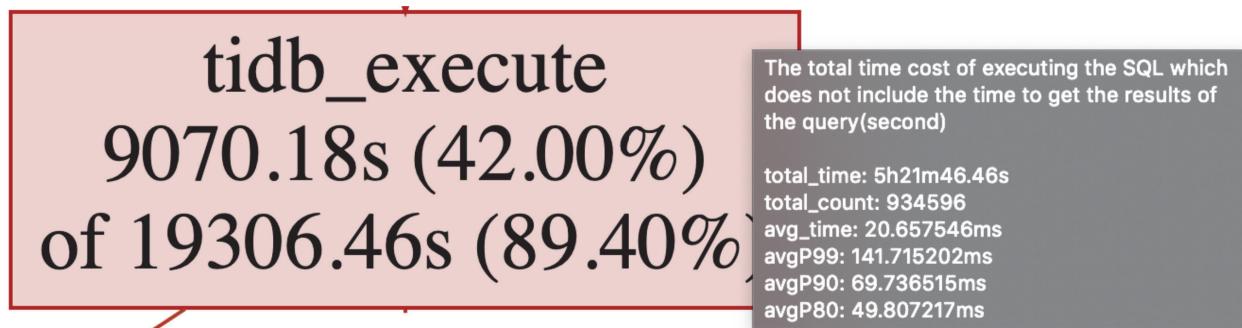


Figure 372: tidb_execute node example2

The text information displayed in the image above is the description of the metric node, including the total duration, the total times, the average duration, and the average duration

P99, P90, and P80.

The parent-child relations between nodes

Taking the `tidb_execute` metric node as an example, this section introduces a metric's child nodes.

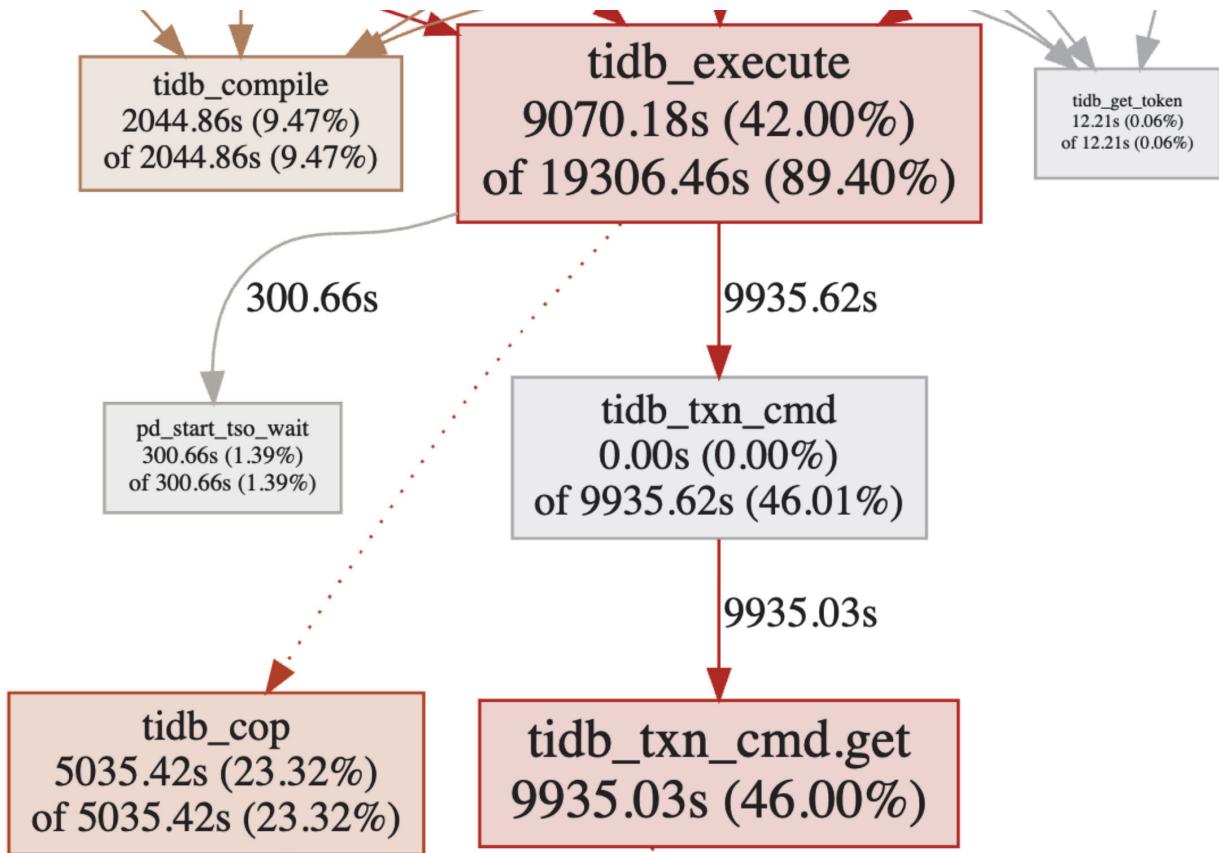


Figure 373: `tidb_execute` node relation example1

From the graph above, you can see the two child nodes of `tidb_execute`:

- `pd_start_tso_wait`: The total duration of waiting for the transaction's `start_tso`, which is 300.66 seconds.
- `tidb_txn_cmd`: The total duration of TiDB executing the relevant transaction commands, which is 9935.62 seconds.

In addition, `tidb_execute` also has a dotted arrow pointing to the `tidb_cop` box area, which indicates as follows:

`tidb_execute` includes the duration of the `tidb_cop` metric, but `cop` requests might be executed concurrently. For example, the `execute` duration of performing `join` queries on two tables is 60 seconds, during which table scan requests are concurrently executed on

the joined two tables. If the execution durations of cop requests are respectively 40 seconds and 30 seconds, the total duration of cop requests are 70 seconds. However, the `execute` duration is only 60 seconds. Therefore, if the duration of a parent node does not completely include the duration of a child node, the dotted arrow is used to point to the child node.

Note:

When a node have a dotted arrow pointing to its child node, the duration of this node itself is inaccurate. For example, in the `tidb_execute` node, the duration of the node itself is 9070.18 seconds ($9070.18 = 19306.46 - \rightarrow 300.66 - 9935.62$). In this equation, the duration of the `tidb_cop` child node is not calculated into the duration of `tidb_execute`'s child nodes. But in fact, this is not true. 9070.18 seconds, the duration of `tidb_execute` itself, includes a part of the `tidb_cop` duration, and the duration of this part cannot be determined.

`tidb_kv_request` and its parent nodes

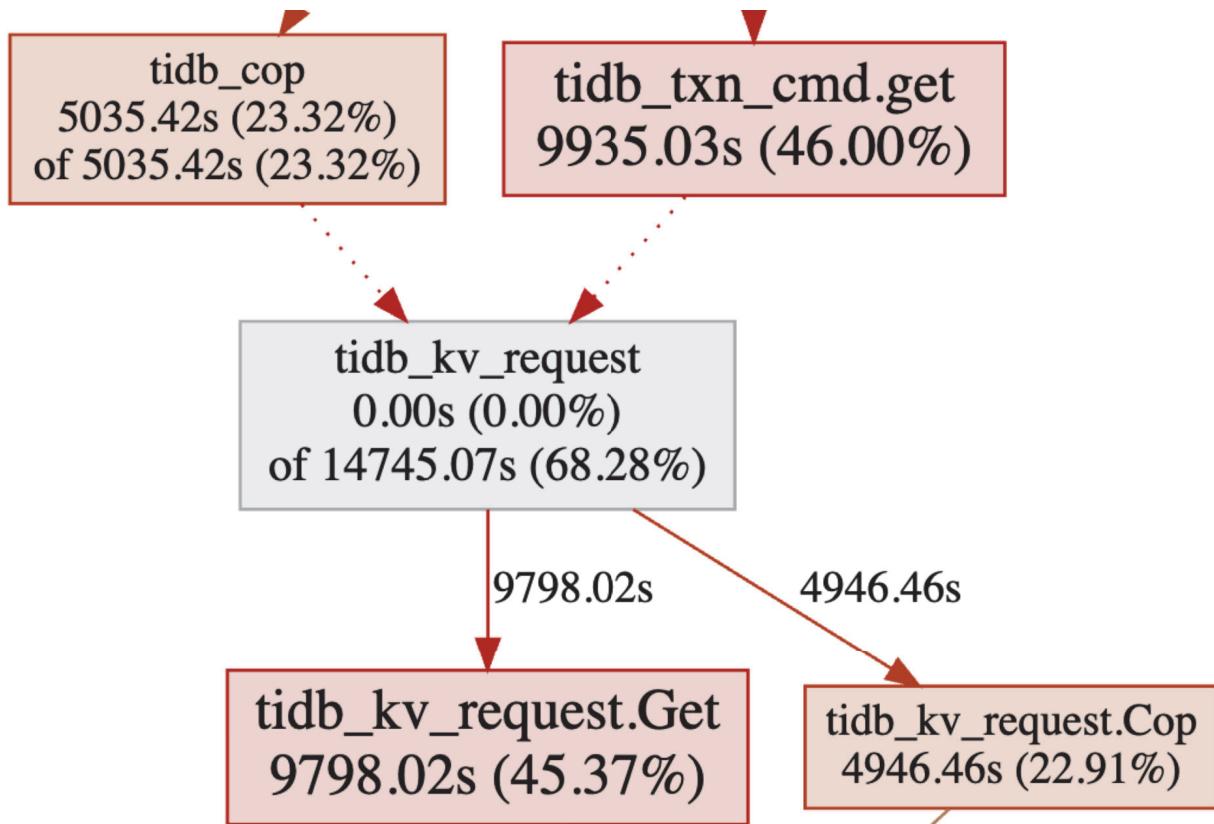


Figure 374: `tidb_execute` node relation example2

`tidb_cop` and `tidb_txn_cmd.get`, the parent nodes of `tidb_kv_request`, both have dotted arrows pointing to `tidb_kv_request`, which indicates as follows:

- The duration of `tidb_cop` includes a part of `tidb_kv_request`'s duration.
- The duration of `tidb_txn_cmd.get` also includes a part of `tidb_kv_request`'s duration.

However, it is hard to determine how much duration of `tidb_kv_request` is included in `tidb_cop`.

- `tidb_kv_request.Get`: The duration of TiDB sending the `Get` type key-value requests.
- `tidb_kv_request.Cop`: The duration of TiDB sending the `Cop` type key-value requests.

`tidb_kv_request` does not include `tidb_kv_request.Get` and `tidb_kv_request.Cop` nodes as its child nodes, but consists of the latter two nodes. The name prefix of the child node is the name of the parent node plus `.xxx`, which means that the child node is the sub-class of the parent node. You can understand this case in the following way:

The total duration of TiDB sending key-value requests is 14745.07 seconds, during which the key-value requests for the `Get` and `Cop` types respectively consume 9798.02 seconds and 4946.46 seconds.

11.6.1.8 SQL Statements Analysis

11.6.1.8.1 SQL Statements Page of TiDB Dashboard

The SQL statements page shows the execution status of all SQL statements in the cluster. This page is often used to analyze the SQL statement whose total or single execution time is long.

On this page, SQL queries with a consistent structure (even if the query parameters are inconsistent) are classified as the same SQL statement. For example, both `SELECT * FROM employee WHERE id IN (1, 2, 3)` and `select * from EMPLOYEE where ID in (4, 5)` are classified as the same `select * from employee where id in (...)` SQL statement.

Access the page

You can use one of the following two methods to access the SQL statement summary page:

- After logging into TiDB Dashboard, click **SQL Statements** on the left navigation menu:

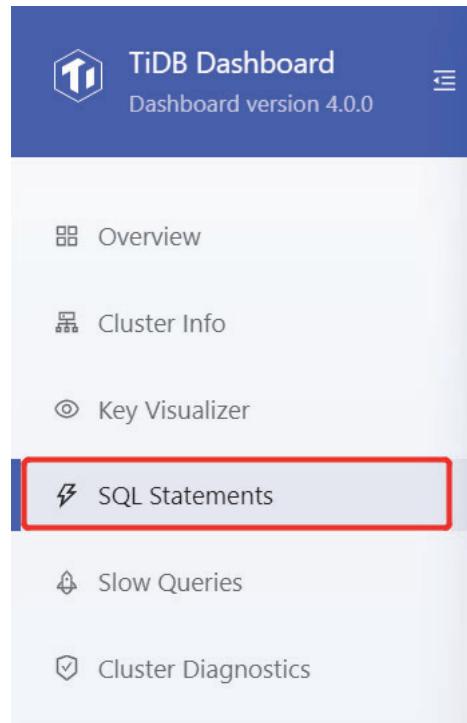


Figure 375: Access SQL statement summary page

- Visit <http://127.0.0.1:2379/dashboard/#/statement> in your browser. Replace 127.0.0.1:2379 with the actual PD instance address and port.

All the data shown on the SQL statement summary page are from the TiDB statement summary tables. For more details about the tables, see [TiDB Statement Summary Tables](#).

Note:

In the **Mean Latency** column of the SQL statement summary page, the blue bar indicates the average execution time. If there is a yellow line on the blue bar for an SQL statement, the left and right sides of the yellow line respectively represent the minimum and maximum execution time of the SQL statement during the recent data collection cycle.

Change Filters

On the top of the SQL statement summary page, you can modify the time range of SQL executions to be displayed. You can also filter the list by database in which SQL statements are executed, or by SQL types. The following image shows all SQL executions over the recent data collection cycle (recent 30 minutes by default).

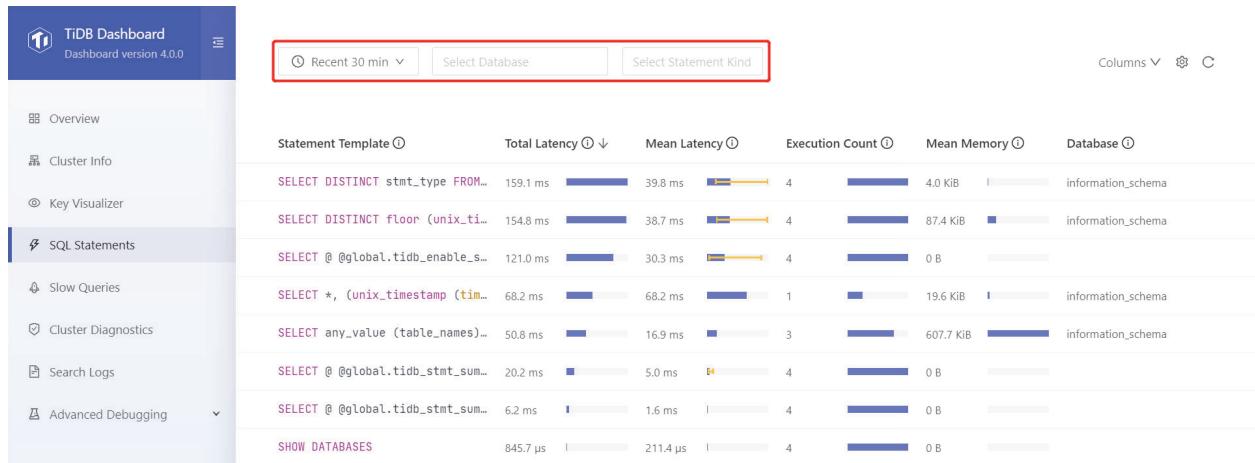


Figure 376: Modify filters

Display More Columns

Click **Columns** on the page and you can choose to see more columns. You can move your mouse to the (i) icon at the right side of a column name to view the description of this column:

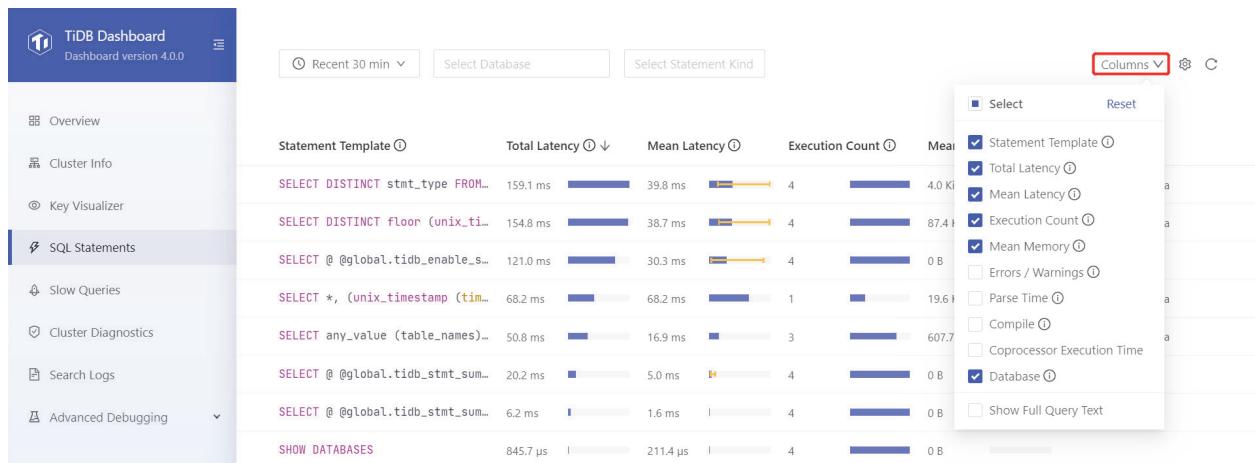


Figure 377: Choose columns

Sort by Column

By default, the list is sorted by **Total Latency** from high to low. Click on different column headings to modify the sorting basis or switch the sorting order:

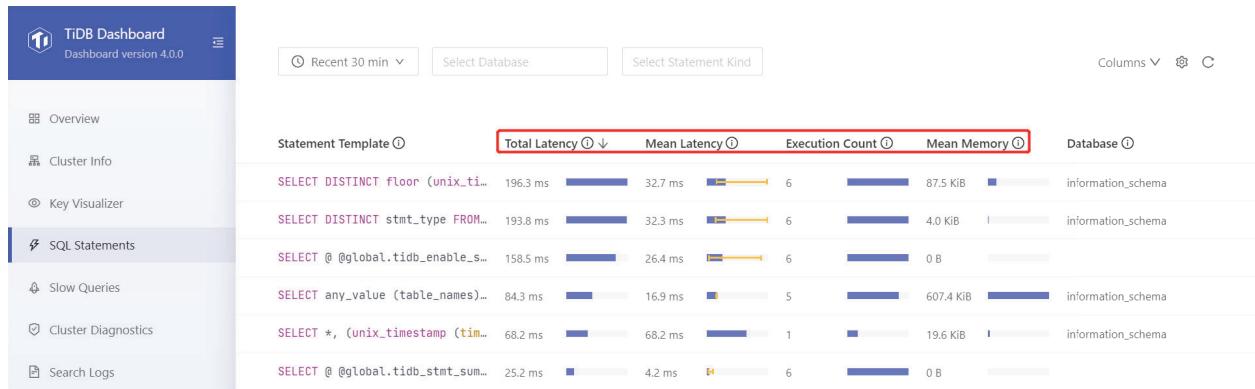


Figure 378: Modify list sorting

Change Settings

On the list page, click the **Settings** button on the top right to change the settings of the SQL statements feature:

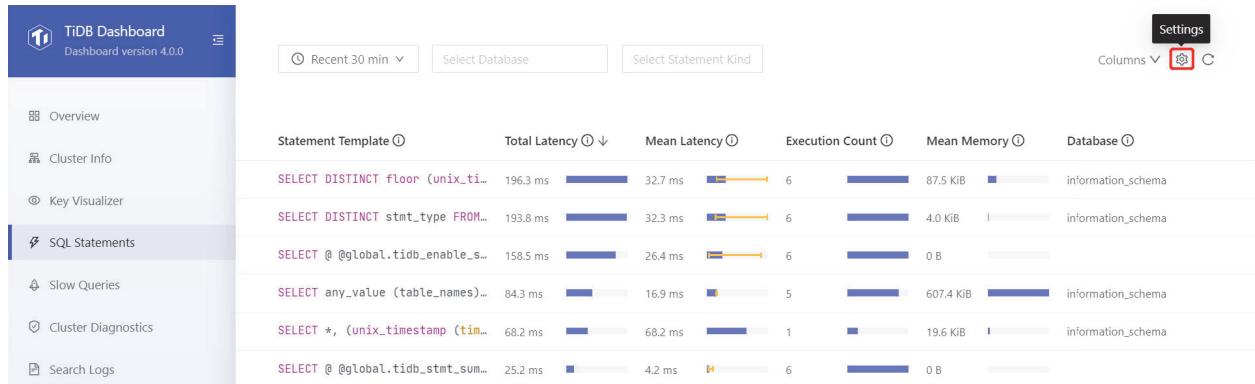


Figure 379: Settings entry

After clicking the **Settings** button, you can see the following setting dialog box:

Settings

X

Enable



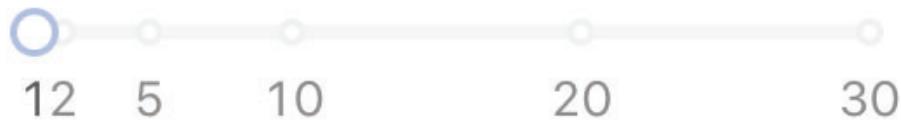
Collect interval

30 min



Data retain duration

1 day



Save

Cancel

Figure 380: Settings
2152

On the setting page, you can disable or enable the SQL statements feature. When the SQL statements feature is enabled, you can modify the following settings:

- Collect interval: The length of period for each SQL statement analysis, which is 30 minutes by default. The SQL statements feature summarizes and counts all SQL statements within a period of time. If the period is too long, the granularity of the summary is coarse, which is not good for locating problems; if the period is too short, the granularity of the statistics is fine, which is good for locating problems, but this will result in more records and more memory usage within the same data retention duration. Therefore, you need to adjust this value based on the actual situation, and properly lower this value when locating problems.
- Data retain duration: The retention duration of summary information, which is 1 day by default. Data retained longer than this duration will be deleted from system tables.

See [Configurations of Statement Summary Tables](#) for details.

Note:

- Because the statement system table is only stored in memory, after the SQL Statements feature is disabled, the data in the system table will be cleared.
- The values of `Collect interval` and `retain duration` affect the memory usage, so it is recommended to adjust these values according to the actual situation. The value of `retain duration` should not be set too large.

11.6.1.8.2 Statement Execution Details of TiDB Dashboard

Click any item in the list to enter the detail page of the SQL statement to view more detailed information. This information includes the following parts:

- The overview of SQL statements, which includes the SQL template, the SQL template ID, the current time range of displayed SQL executions, the number of execution plans and the database in which the SQL statement is executed (see area 1 in the image below).
- The execution plan list: If the SQL statement has multiple execution plans, this list is displayed. You can select different execution plans, and the details of the selected plans are displayed below the list. If there is only one execution plan, the list is not displayed (see area 2 below).
- Execution detail of plans, which displays the detailed information of the selected execution plans. See [Execution plan in details](#) (area 3 in the image below).

[List](#) Statement Information

SQL Template ⓘ Expand Copy ⓘ
SELECT * FROM t1 WHERE t_num BETWEEN ? AND ?

SQL Template ID ⓘ Copy ⓘ
 0369920d9e7d27f1972da991c66a352d6be2722f93bc7c445d8218c6c3fa993e Selected Time Range
 Today at 1:30 PM ~ Today at 2:30 PM

Execution Plans
 3 Execution Database ⓘ Copy ⓘ test

ⓘ There are multiple execution plans for this kind of SQL. You can choose to view one or multiple of them.

Plan ID ⓘ	Total Latency ⓘ	Mean Latency ⓘ	Execution Count ⓘ	Mean Memory ⓘ
adc558fb558d251f4322788bc70a8ae5...	27.8 ms	9.3 ms	3	104.6 ...
f965f00dd5dcfa2b33e30cadf5a47bdaed...	16.9 ms	16.9 ms	1	236.3 ...
0797ea336a604540de3a6cf562a22b3b...	1.9 ms	1.9 ms	1	9.2 KiB

Execution Detail of All Plans

SQL Sample Expand Copy ⓘ
SELECT * FROM t1 WHERE t_num BETWEEN 20 AND 21

Execution Plan Collapse Copy ⓘ
 IndexReader_6 root 5.999999999999999 index:IndexRangeScan_5
 └─IndexScan_5 cop 5.999999999999999 table:t1, index:t_num(t_num), range:[20,21], keep order:false

Basic Time Coprocessor Read Transaction Slow Query

Name	Value	Description
Table Names	test.t1	
Index Name	t1:t_num	The name of the used index
First Seen	Today at 1:53 PM	

Figure 381: Details

Execution details of plans

The execution detail of plans includes the following information:

- SQL sample: The text of a certain SQL statement that is actually executed corresponding to the plan. Any SQL statement that has been executed within the time range might be used as a SQL sample.
- Execution plan: For details of the execution plan, see [Understand the Query Execution Plan](#). If multiple execution plans are selected, only (any) one of them is displayed.
- For basic information, execution time, Coprocessor read, transaction, and slow query of the SQL statement, you can click the corresponding tab titles to switch among different information.

Execution Detail of All Plans

SQL Sample [Expand](#) [Copy](#) [?](#)

```
SELECT * FROM t1 WHERE t_num BETWEEN 0 AND 999999999
```

Execution Plan [Expand](#) [Copy](#) [?](#)

```
TableReader_7 root 10000 data:Selection_6 └─Selection_6 cop 10000 ge(test.t1.t_num, 0), le(test.t1.t_num, 99)
```

[Basic](#) [Time](#) [Coprocessor Read](#) [Transaction](#) [Slow Query](#)

Name	Value	Description
Table Names	test.t1	
Index Name		The name of the used index
First Seen	Today at 1:53 PM	
Last Seen	Today at 2:00 PM	
Execution Count	5	Total execution count for this kind of SQL
Total Latency	46.7 ms	Total execution time for this kind of SQL
Execution User	root	The user that executes the SQL (sampled)

Figure 382: Execution details of plans

Basic Tab

The basic information of a SQL execution includes the table names, index name, execution count, and total latency. The **Description** column provides detailed description of each field.

Basic	Time	Coprocessor Read	Transaction	Slow Query
Name	Value	Description		
Table Names	test.t1			
Index Name		The name of the used index		
First Seen	Today at 1:53 PM			
Last Seen	Today at 2:00 PM			
Execution Count	5	Total execution count for this kind of SQL		
Total Latency	46.7 ms	Total execution time for this kind of SQL		
Execution User	root	The user that executes the SQL (sampled)		
Total Errors	0			
Total Warnings	0			
Mean Memory	111.9 KiB	Memory usage of single SQL		
Max Memory	236.3 KiB	Maximum memory usage of single SQL		

Figure 383: Basic information

Time Tab

Click the **Time** tab, and you can see how long each stage of the execution plan lasts.

Note:

Because some operations might be performed in parallel within a single SQL statement, the cumulative duration of each stage might exceed the actual execution time of the SQL statement.

Basic	Time	Coprocessor Read	Transaction	Slow Query
Name	Time	Description		
Parse	69.7 µs			Time consumed when parsing the SQL statement
Compile	331.4 µs			Time consumed when optimizing the SQL state...
Coprocessor Wait	0 ns			
Coprocessor Execution	400.0 µs			
Backoff Retry	0 ns			
Get Commit Ts	0 ns			
Local Latch Wait	0 ns			
Resolve Lock	0 ns			
Prewrite	0 ns			
Commit	0 ns			
Commit Backoff Retry	0 ns			
Query	9.3 ms			

Figure 384: Execution time

Coprocessor Read Tab

Click the **Coprocessor Read** tab, and you can see information related to Coprocessor read.

Basic	Time	Coprocessor Read	Transaction	Slow Query
Name	Value	Description		
Total Coprocessor Tasks	6			
Mean Visible Versions per SQL	6.0 K			
Max Visible Versions per SQL	10.0 K			
Mean Meet Versions per SQL	6.0 K	Meet versions contains overwritten or deleted v...		
Max Meet Versions per SQL	10.0 K			

Figure 385: Coprocessor read

Transaction Tab

Click the **Transaction** tab, and you can see information related to execution plans and transactions, such as the average number of written keys or the maximum number of written keys.

Basic	Time	Coprocessor Read	Transaction	Slow Query
Name	Value	Description		
Mean Affected Rows	0			
Total Backoff Count	0			
Mean Written Keys	0			
Max Written Keys	0			
Mean Written Data Size	0 B			
Max Written Data Size	0 B			
Mean Prewrite Regions	0			
Max Prewrite Regions	0			
Mean Transaction Retries	0			
Max Transaction Retries	0			

Figure 386: Transaction

Slow Query Tab

If an execution plan is executed too slowly, you can see its associated slow query records under the **Slow Query** tab.

Basic	Time	Coprocessor Read	Transaction	Slow Query
SQL ⓘ	Finish Time ⓘ ↓	Latency ⓘ	Max Memory ⓘ	
ANALYZE TABLE t;	Today at 1:19 PM	336.5 ms	0 B	

Figure 387: Slow Query

The information displayed in this area has the same structure with the slow query page. See [TiDB Dashboard Slow Query Page](#) for details.

11.6.1.9 Slow Queries Page of TiDB Dashboard

On the Slow Queries page of TiDB Dashboard, you can search and view all slow queries in the cluster.

By default, SQL queries with an execution time of more than 300 milliseconds are considered as slow queries. These queries are recorded in the `slow query logs` and can be searched via TiDB Dashboard. You can adjust the threshold of slow queries through the `tidb_slow_log_threshold` session variable or the `slow-threshold` TiDB parameter.

Note:

If the slow query log is disabled, this feature will be unavailable. The slow query log is enabled by default, and you can enable or disable the slow query log through the `enable-slow-log` TiDB configuration item.

11.6.1.9.1 Access the page

You can use one of the following two methods to access the slow query page:

- After logging into TiDB Dashboard, click **Slow Queries** on the left navigation menu:

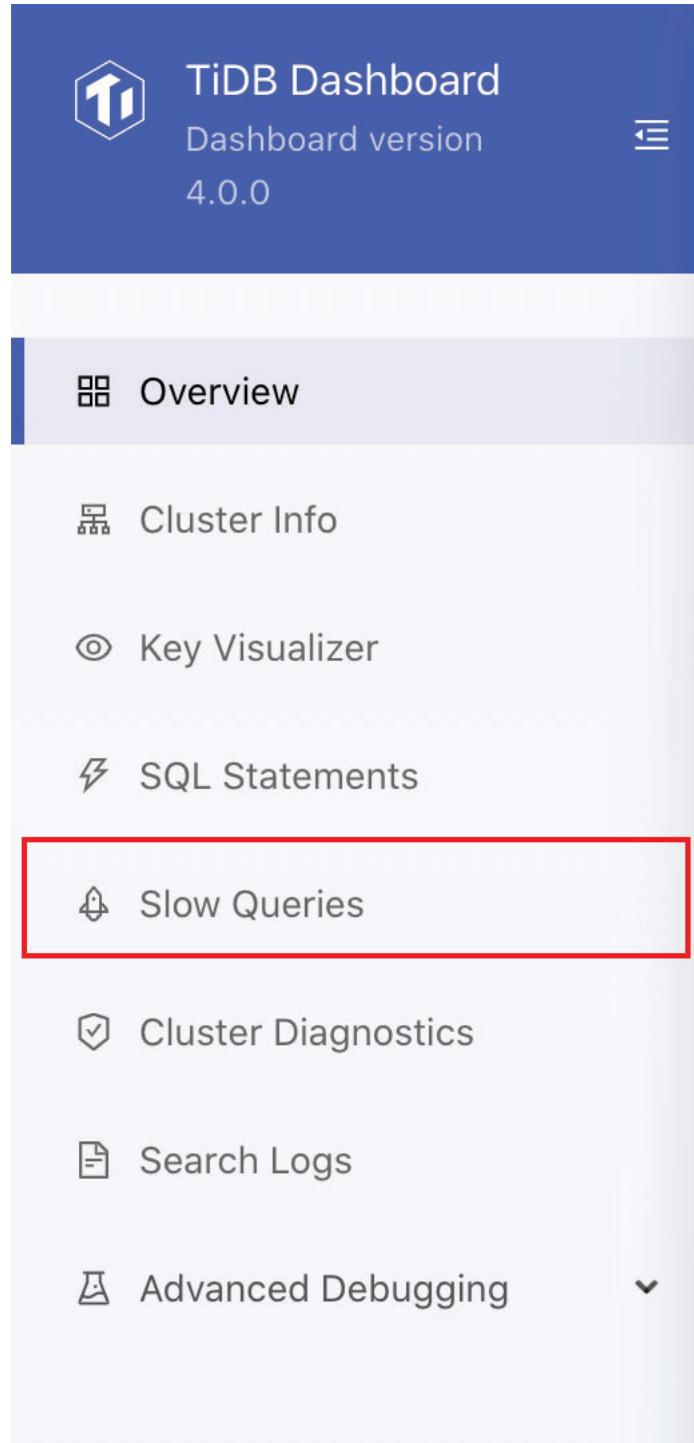


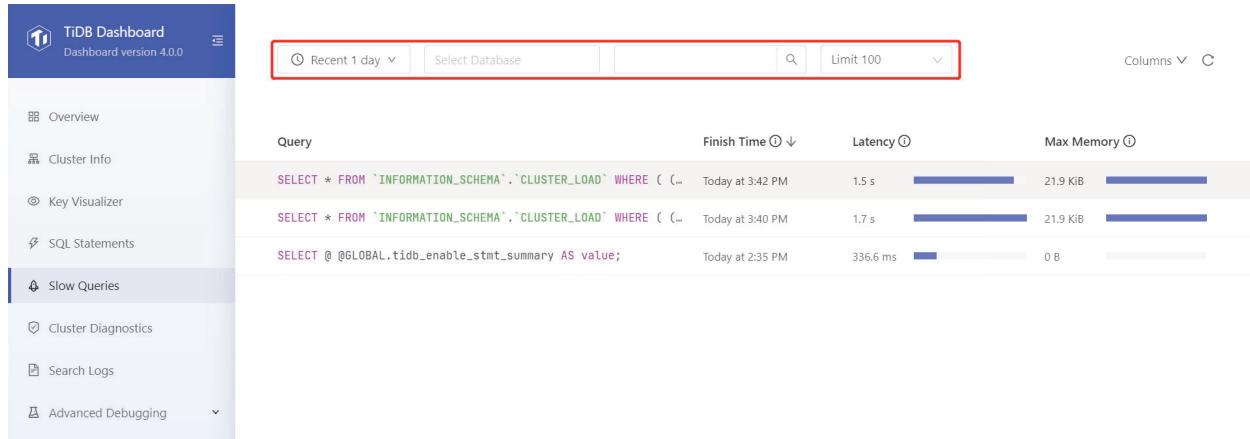
Figure 388: Access slow query page

- Visit http://127.0.0.1:2379/dashboard/#/slow_query in your browser. Replace 127.0.0.1:2379 with the actual PD address and port.

All data displayed on the slow query page comes from TiDB slow query system tables and slow query logs. See [slow query logs](#) for details.

Change Filters

You can filter slow queries based on the time range, the related database, SQL keywords, SQL types, the number of slow queries to be displayed. In the image below, 100 slow queries over the recent 30 minutes are displayed by default.



Query	Finish Time ⓘ ↓	Latency ⓘ	Max Memory ⓘ
SELECT * FROM `INFORMATION_SCHEMA`.`CLUSTER_LOAD` WHERE ((...	Today at 3:42 PM	1.5 s	21.9 KIB
SELECT * FROM `INFORMATION_SCHEMA`.`CLUSTER_LOAD` WHERE ((...	Today at 3:40 PM	1.7 s	21.9 KIB
SELECT @@GLOBAL.tidb_enable_stmt_summary AS value;	Today at 2:35 PM	336.6 ms	0 B

Figure 389: Modify list filters

Display More Columns

Click **Columns** on the page and you can choose to see more columns. You can move your mouse to the (i) icon at the right side of a column name to view the description of this column:

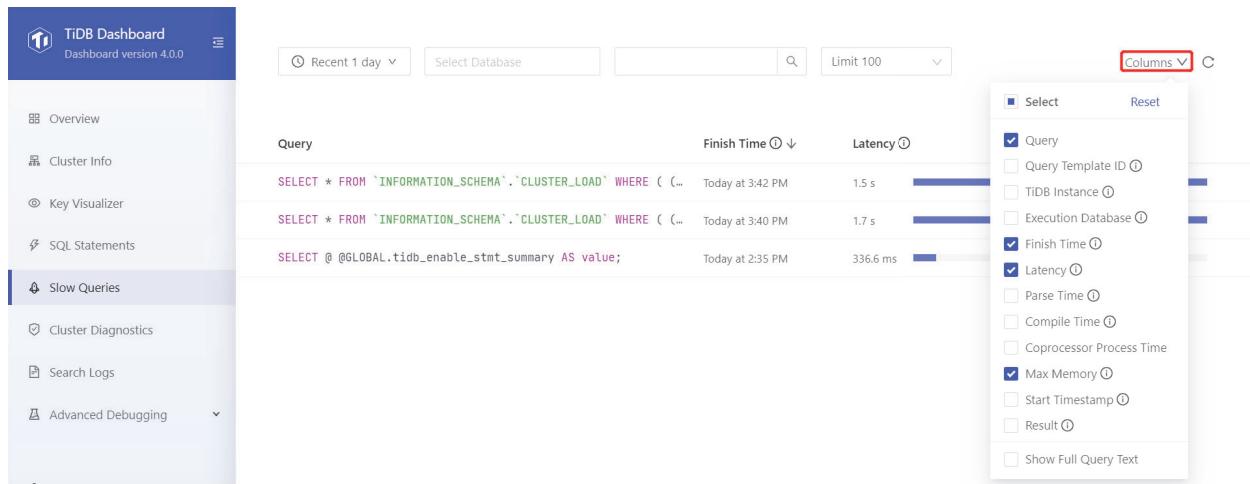
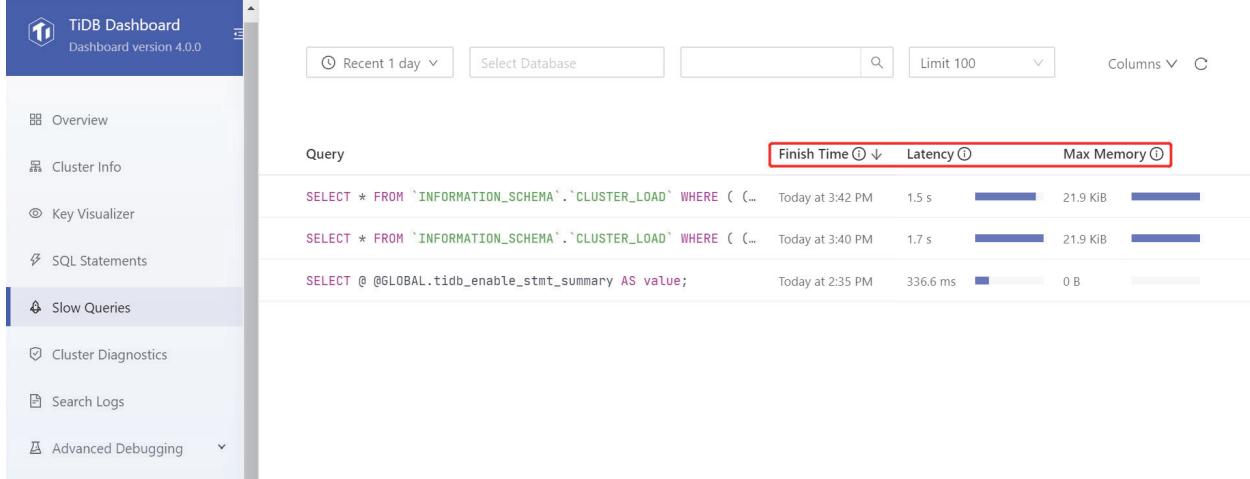


Figure 390: Show more columns

Sort by Column

By default, the list is sorted by **Finish Time** in the descending order. Click column headings to sort by the column or switch the sorting order:



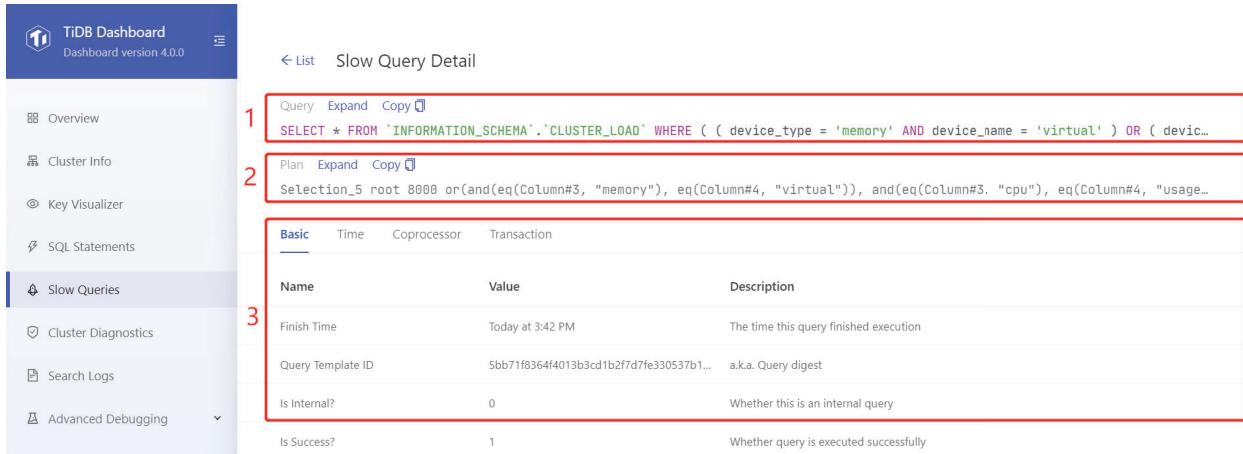
Query	Finish Time ⓘ ↓	Latency ⓘ	Max Memory ⓘ
SELECT * FROM `INFORMATION_SCHEMA`.`CLUSTER_LOAD` WHERE ((device_type = 'memory' AND device_name = 'virtual') OR (device_type = 'cpu' AND device_name = 'usage'))	Today at 3:42 PM	1.5 s	<div style="width: 80%;">80%</div> 21.9 KIB
SELECT * FROM `INFORMATION_SCHEMA`.`CLUSTER_LOAD` WHERE ((device_type = 'memory' AND device_name = 'virtual') OR (device_type = 'cpu' AND device_name = 'usage'))	Today at 3:40 PM	1.7 s	<div style="width: 80%;">80%</div> 21.9 KIB
SELECT @@GLOBAL.tidb_enable_stmt_summary AS value;	Today at 2:35 PM	336.6 ms	<div style="width: 5%;">5%</div> 0 B

Figure 391: Modify sorting basis

11.6.1.9.2 View execution details

Click any item in the list to display detailed execution information of the slow query, including:

- Query: The text of the SQL statement (see area 1 in the image below);
- Plan: The execution plan of the slow query. See [Understand the Query Execution Plan](#) to learn how to read the execution plan (see area 2 in the image below);
- Other sorted SQL execution information (see area 3 in the image below).



Slow Query Detail

1 Query Expand Copy ⓘ
SELECT * FROM `INFORMATION_SCHEMA`.`CLUSTER_LOAD` WHERE ((device_type = 'memory' AND device_name = 'virtual') OR (device_type = 'cpu' AND device_name = 'usage'))

2 Plan Expand Copy ⓘ
Selection_5 root 8000 or(and(eq(Column#3, "memory"), eq(Column#4, "virtual")), and(eq(Column#3, "cpu"), eq(Column#4, "usage")))

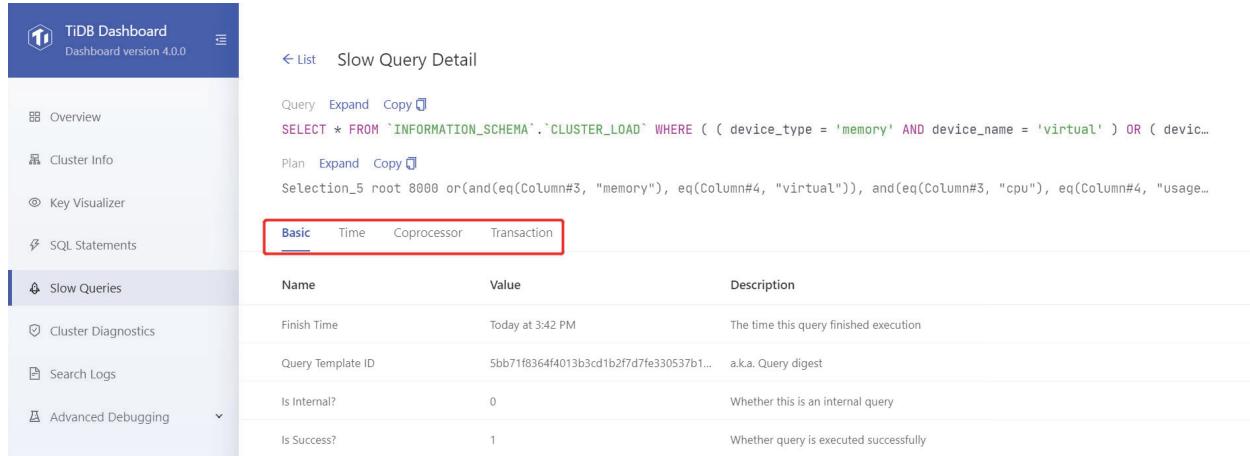
3 Basic Time Coprocessor Transaction

Name	Value	Description
Finish Time	Today at 3:42 PM	The time this query finished execution
Query Template ID	5bb71f8364f4013b3cd1b2f7d7fe330537b1...	a.k.a. Query digest
Is Internal?	0	Whether this is an internal query
Is Success?	1	Whether query is executed successfully

Figure 392: View execution details

Click the **Expand** link to show the detailed information of an item. Click the **Copy** link to copy the detailed information to the clipboard.

Click the corresponding tab titles to switch information of different sorted SQL executions.



Name	Value	Description
Finish Time	Today at 3:42 PM	The time this query finished execution
Query Template ID	5bb71f8364f4013b3cd1b2f7d7fe330537b1...	a.k.a. Query digest
Is Internal?	0	Whether this is an internal query
Is Success?	1	Whether query is executed successfully

Figure 393: Show different sorted execution information

11.6.1.10 Cluster Diagnostics

11.6.1.10.1 TiDB Dashboard Cluster Diagnostics Page

Warning:

Diagnostics in TiDB Dashboard is still an experimental feature. It is **NOT** recommended that you use it in the production environment.

The cluster diagnostics feature in TiDB Dashboard diagnoses the problems that might exist in a cluster within a specified time range, and summarizes the diagnostic results and the cluster-related load monitoring information into a diagnostic report. This diagnostic report is in the form of a web page. You can browse the page offline and circulate this page link after saving the page from a browser.

Note:

The cluster diagnostics feature depends on Prometheus deployed in the cluster. For details about how to deploy this monitoring component, see the

TiUP deployment document. If no monitoring component is deployed in the cluster, the generated diagnostic report will indicate a failure.

Access the page

You can use one of the following methods to access the cluster diagnostics page:

- After logging into TiDB Dashboard, click **Cluster Diagnostics** on the left navigation menu:

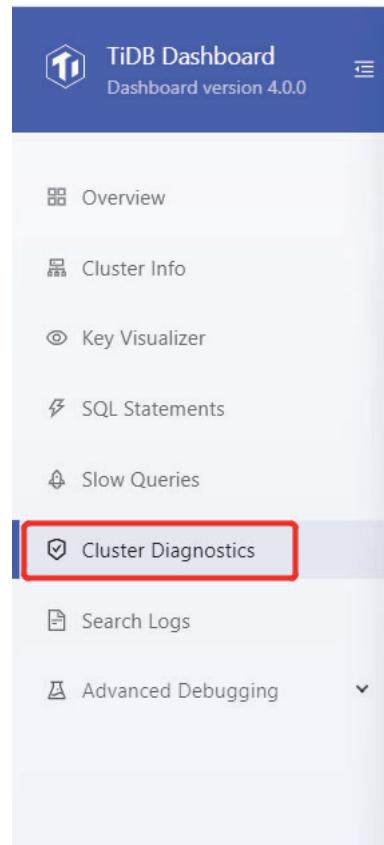


Figure 394: Access Cluster Diagnostics page

- Visit `http://127.0.0.1:2379/dashboard/#/diagnose` in your browser. Replace `127.0.0.1:2379` with the actual PD address and port number.

Generate diagnostic report

To diagnose a cluster within a specified time range and check the cluster load, you can take the following steps to generate a diagnostic report:

1. Set the **Range Start Time**, such as 2020-05-21 14:40:00.
2. Set the **Range Duration**, such as 10 min.
3. Click **Start**.

New Diagnostic Report

* Range Start Time:
* Range Duration:
Compare by Baseline:
Start

Report ID	May 2020							14:40:00			Range	
	Su	Mo	Tu	We	Th	Fr	Sa	14	40	00		
23610781-8789-4325-9	26	27	28	29	30	1	2	15	41	01	May 21, 2020 2:30 PM ~ May 21, 2020 2:35 PM	
887a74ff-893c-41bb-8	3	4	5	6	7	8	9	16	42	02	Finish	
	10	11	12	13	14	15	16	17	43	03	Last Tuesday 11:43 AM ~ Last Tuesday 11:53 AM	
	17	18	19	20	21	22	23	18	44	04		
	24	25	26	27	28	29	30	19	45	05		
	31	1	2	3	4	5	6	20	46	06		
								21	47	07		
	Now							Ok				

Figure 395: Generate diagnostic report

Note:

It is recommended that the **Range Duration** of the report is between 1 minute and 60 minutes. This **Range Duration** cannot exceed 60 minutes.

The steps above generate a diagnostic report for the time range from 2020-05-21 → 14:40:00 to 2020-05-21 14:50:00. After clicking **Start**, you can see the interface below. **Progress** is the progress bar of the diagnostic report. After the report is generated, click **View Full Report**.

[← New Diagnostic Report](#) Diagnostic Status [View Full Report](#)

Range Start Time	May 21, 2020 2:40 PM
Range End Time	May 21, 2020 2:50 PM
Progress	<div style="width: 100%;"><div style="width: 100%;"> </div></div>

Figure 396: Report progress

Generate comparison report

If a system exception occurs at a certain point, for example, QPS jitter or higher latency, a diagnostic report can be generated. Particularly, this report compares the system in the abnormal time range with the system in the normal time range. For example:

- Abnormal time range: 2020-05-21 14:40:00-2020-05-21 14:45:00. Within this time range, the system is abnormal.
- Normal time range: 2020-05-21 14:30:00 - 2020-05-21 14:35:00. Within this time range, the system is normal.

You can take the following steps to generate a comparison report for the two time ranges above:

1. Set the **Range Start Time**, which is the start time of the range in which the system becomes abnormal, such as 2020-05-21 14:40:00.
2. Set the **Range Duration**. Generally, this duration is the duration of system anomalies, such as 5 minutes.
3. Enable **Compare by baseline**.
4. Set the **Baseline Range Start Time**, which is the start time of the range (to be compared with) in which the system is normal, such as 2020-05-21 14:30:00.
5. Click **Start**.

New Diagnostic Report

* Range Start Time: * Range Duration: Compare by Baseline: * Baseline Range Start Time: [Start](#)

Report ID	Diagnose At	Status	Range
2f4de11e-2d1f-4f30-940f-5e0e697f9afa	Today at 11:54 AM	Finish	May 21, 2020 2:40 PM ~ May 21, 2020 2:50 PM
23610781-8789-4325-9189-3482fcbe35a1	Today at 11:47 AM	Finish	May 21, 2020 2:30 PM ~ May 21, 2020 2:35 PM
887a74ff-893c-41bb-812a-6295f37ae592	Today at 11:43 AM	Finish	Last Tuesday 11:43 AM ~ Last Tuesday 11:53 AM

History report list

May 2020

Su	Mo	Tu	We	Th	Fr	Sa	14	30	00
26	27	28	29	30	1	2	15	31	01
3	4	5	6	7	8	9	16	32	02
10	11	12	13	14	15	16	17	33	03
17	18	19	20	21	22	23	18	34	04
24	25	26	27	28	29	30	19	35	05
31	1	2	3	4	5	6	20	36	06
							21	37	07

Now [ok](#)

Figure 397: Generate comparison report

Then wait for the report to be generated and click **View Full Report**.

In addition, the historical diagnostic report is displayed in the list on the main page of the diagnostic report. You can click to view these historical reports directly.

11.6.1.10.2 TiDB Dashboard Diagnostic Report

This document introduces the content of the diagnostic report and viewing tips. To access the cluster diagnostic page and generate reports, see [TiDB Dashboard Cluster Diagnostics Page](#).

[View report](#)

The diagnostic report consists of the following parts:

- Basic information: Includes the time range of the diagnostic report, hardware information of the cluster, the version information of cluster topology.
- Diagnostic information: Shows the results of automatic diagnostics.
- Load information: Includes CPU, memory and other load information of the server, TiDB, PD, or TiKV.
- Overview information: Includes the consumed time and error information of each TiDB, PD, or TiKV module.
- TiDB/PD/TiKV monitoring information: Includes monitoring information of each component.
- Configuration information: Includes configuration information of each component.

An example of the diagnostic report is as follows:

Total Time Consume								
The table contain the event time consume in TiDB/TiKV/PD. METRIC_NAME is the event name; LABEL is the event label, such as instance, event type ...; TIME_RATIO is the TOTAL_TIME of this event devide by the TOTAL_TIME of upper event which TIME_RATIO is 1; TOTAL_TIME is the total time cost of this event; TOTAL_COUNT is the total count of this event; P999 is the max time of 0.999 quantile; P99 is the max time of 0.99 quantile; P90 is the max time of 0.90 quantile; P80 is the max time of 0.80 quantile;								
METRIC_NAME	LABEL	TIME_RATIO	TOTAL_TIME	TOTAL_COUNT	P999	P99	P90	P80
tidb_query	 expand	1	23223.86	459625	63.54	45.55	0.1	0.06
tidb_get_token	 fold	0.000001	0.03	458888	0.00004	0.000001	0.000001	0.000001
I-- tidb_get_token	10.0.1.13:10080	0.000001	0.03	458888	0.00004	0.000001	0.000001	0.000001

Figure 398: Sample report

In the image above, **Total Time Consume** in the top blue box is the report name. The information in the red box below explains the content of this report and the meaning of each field in the report.

In this report, some small buttons are described as follows:

- **i** icon: You can move your mouse to the **i** icon to see the explanatory note of the row.

- **expand**: Click **expand** to see details about this monitoring metric. For example, the detailed information of `tidb_get_token` in the image above includes the monitoring information of each TiDB instance's latency.
- **collapse**: Contrary to **expand**, the button is used to fold detailed monitoring information.

All monitoring metrics basically correspond to those on the TiDB Grafana monitoring dashboard. After a module is found to be abnormal, you can view more monitoring information on the TiDB Grafana.

In addition, the `TOTAL_TIME` and `TOTAL_COUNT` metrics in this report are monitoring data read from Prometheus, so calculation inaccuracy might exist in their statistics.

Each part of this report is introduced as follows.

Basic information

Diagnostics Time Range

The time range for generating the diagnostics report includes the start time and end time.

Report Time Range	
START_TIME	END_TIME
2020-05-21 06:40:00	2020-05-21 06:50:00

Figure 399: Report time range

Cluster Hardware Info

Cluster Hardware Info includes information such as CPU, memory, and disk of each server in the cluster.

Cluster Hardware					
HOST	INSTANCE	CPU_CORES	MEMORY (GB)	DISK (GB)	UPTIME (DAY)
10.0.1.14	tiflash*1	20/40	125.64	nvme01: 498.638	2.9132028513567314
10.0.1.11	pd*1	20/40	125.64	sda1: 965.834	2.921851028772416
10.0.1.12	tikv*1	20/40	125.64	nvme0n1: 122.78 sda1: 965.834	2.9197623066052247
10.0.1.13	tidb*1	20/40	125.64	sda1: 498.638	2.917983893669314

Figure 400: Cluster hardware report

The fields in the table above are described as follows:

- **HOST**: The IP address of the server.
- **INSTANCE**: The number of instances deployed on the server. For example, `pd * 1` means that this server has 1 PD instance deployed; `tidb * 2 pd * 1` means that this server has 2 TiDB instances and 1 PD instance deployed.

- **CPU_CORES**: Indicates the number of CPU cores (physical cores or logical cores) of the server.
- **MEMORY**: Indicates the memory size of the server. The unit is GB.
- **DISK**: Indicates the server disk size. The unit is GB.
- **UPTIME**: The uptime of the server. The unit is day.

Cluster Topology Info

The **Cluster_Info** table shows the cluster topology information. The information in this table are from TiDB [information_schema.cluster_info](#) system table.

Cluster Info

TYPE	INSTANCE	STATUS_ADDRESS	VERSION	GIT_HASH	START_TIME	UPTIME
pd	10.0.1.11:2379	10.0.1.11:2379	4.1.0-alpha	a80b99ef0d70807d106bcd1b7c6e97a118679c7e	2020-05-18T13:47:01Z	65h40m54.520385474s
tidb	10.0.1.13:4000	10.0.1.13:10080	4.0.0-beta.2	ac30f5322e253125002663ad7c789807acf9305	2020-05-18T13:47:10Z	65h40m45.520383242s
tiflash	10.0.1.14:3930	10.0.1.14:20292	v4.1.0-alpha-37-gacf3f1673d	acf3f673dfbc3e6ffff0dda575760670993fefa6	2020-05-18T13:47:18Z	65h40m37.520390633s
tikv	10.0.1.12:20160	10.0.1.12:20180	4.1.0-alpha	6b09cadbf55311ac07fc51e1b43e3b57b54e71f2	2020-05-18T13:47:04Z	65h40m51.520389948s

Figure 401: Cluster info

The fields in the table above are described as follows:

- **TYPE**: The node type.
- **INSTANCE**: The instance address(es), which is a string in the IP:PORT format.
- **STATUS_ADDRESS**: The HTTP API service address.
- **VERSION**: The semantic version number of the corresponding node.
- **GIT_HASH**: Git Commit Hash when compiling the node version, which is used to identify whether the two nodes are absolutely the consistent version.
- **START_TIME**: The start time of the corresponding node.
- **UPTIME**: The uptime of the corresponding node.

Diagnostic information

TiDB has built-in automatic diagnostic results. For the description of each field, see [information_schema.inspection-result](#) system table.

Load Info

Node Load Info

The **Node Load Info** table shows the load information of the server node, including the average value (AVG), maximum value (MAX), minimum value (MIN) of the following metrics of the server within the time range:

- CPU usage (the maximum value is 100%)

- Memory usage
- Disk I/O usage
- Disk write latency
- Disk read latency
- Disk read bytes per second
- Disk write bytes per second
- The number of bytes received by the node network per minute
- The number of bytes sent from the node network per minute
- The number of TCP connections in use by the node
- The number of all TCP connections of the node

Node Load Info

METRIC_NAME	instance	AVG	MAX	MIN
node_cpu_usage expand		40.42%	99.6%	6.98%
node_mem_usage expand		66.9%	96.28%	33.65%
node_disk_io_utilization expand		16%	97%	0%
Disk write latency ⓘ expand		4000 us	50 ms	0 us
Disk read latency ⓘ expand		2000 us	6000 us	0 us
tikv_disk_read_bytes expand		605.980 KB	15.842 MB	0.000 KB
tikv_disk_write_bytes expand		1.129 MB	12.235 MB	0.000 KB
node_network_in_traffic expand		225.706 KB	1.427 MB	0.000 KB
node_network_out_traffic expand		165.807 KB	1.659 MB	0.000 KB
node_tcp_in_use expand		19	47	5
node_tcp_connections expand		30	61	6

Figure 402: Server Load Info report

Instance CPU Usage

The **Instance CPU Usage** table shows the average value (AVG), maximum value (MAX), and minimum value (MIN) of the CPU usage of each TiDB/PD/TiKV process. The maximum CPU usage of the process is $100\% * \text{the number of CPU logical cores}$.

INSTANCE	JOB	AVG	MAX	MIN
172.16.5.40:10089	tidb	1891%	1906%	1878%
172.16.5.40:22151	tikv	517%	571%	459%
172.16.5.40:20151	tikv	498%	562%	444%
172.16.5.40:23151	tikv	352%	399%	311%
172.16.5.40:24799	pd	39%	45%	36%

Figure 403: Instance CPU Usage report

Instance Memory Usage

The **Instance Memory Usage** table shows the average value (AVG), maximum value (MAX), and minimum value (MIN) of memory bytes occupied by each TiDB/PD/TiKV process.

INSTANCE	JOB	AVG	MAX	MIN
172.16.5.40:20151	tikv	9.562 GB	9.605 GB	9.523 GB
172.16.5.40:23151	tikv	9.528 GB	9.574 GB	9.496 GB
172.16.5.40:22151	tikv	9.433 GB	9.498 GB	9.345 GB
172.16.5.40:10089	tidb	904.500 MB	911.660 MB	894.336 MB
172.16.5.40:24799	pd	61.089 MB	61.270 MB	60.980 MB

Figure 404: Instance memory usage report

TiKV Thread CPU Usage

The TiKV Thread CPU Usage table shows the average value (AVG), maximum value (MAX) and minimum value (MIN) of CPU usage of each module thread in TiKV. The maximum CPU usage of the process is $100\% * \text{the thread count of the corresponding configuration}$.

TiKV Thread CPU Usage

METRIC_NAME	INSTANCE	AVG	MAX	MIN	CONFIG_KEY	CURRENT_CONFIG_VALUE
grpc  expand		14.83%	16%	12.96%		
schedule worker  fold		8.76%	9.47%	7.56%		
l-- sched_worker	10.0.1.12:20180	8.76%	9.47%	7.56%	storage.scheduler-worker-pool-size	4
storage read pool  expand		3.24%	3.64%	2.67%		
storage read pool normal  expand		3.22%	3.64%	2.67%		
Async apply  fold		3.14%	3.4%	2.71%		
l-- Async apply	10.0.1.12:20180	3.14%	3.4%	2.71%	raftstore.apply-pool-size	2
raftstore  expand		2.7%	2.93%	2.38%		
rocksdb  expand		0.54%	2.13%	0%		
unified read pool  expand		0.45%	4.24%	0%		
split_check  expand		0.26%	0.84%	0%		
gc  expand		0.04%	0.4%	0%		
storage read pool high  expand		0.02%	0.04%	0%		
snapshot  expand		0.004%	0.02%	0%		
cop normal  						
cop high  						
cop low  						
storage read pool low  expand		0%	0%	0%		
cop  						

Figure 405: TiKV Thread CPU Usage report

In the table above,

- **CONFIG_KEY:** The relevant thread configuration of the corresponding module.
- **CURRENT_CONFIG_VALUE:** The current value of the configuration when the report is generated.

Note:

`CURRENT_CONFIG_VALUE` is the value when the report is generated, not the value within the time range of this report. Currently, some configuration values of historical time cannot be obtained.

TiDB/PD Goroutines Count

The **TiDB/PD Goroutines Count** table shows the average value (AVG), maximum value (MAX), and minimum value (MIN) of the number of TiDB or PD goroutines. If the number of goroutines exceeds 2,000, the concurrency of the process is too high, which affects the overall request latency.

INSTANCE	JOB	AVG	MAX	MIN
172.16.5.40:10089	tidb	1434.33	1473	1394
172.16.5.40:24799	pd	157	157	157

Figure 406: TiDB/PD goroutines count report

Overview information

Time Consumed by Each Component

The **Time Consumed by Each Component** table shows the monitored consumed time and the time ratio of TiDB, PD, TiKV modules in the cluster. The default time unit is seconds. You can use this table to quickly locate which modules consume more time.

METRIC_NAME	LABEL	TIME_RATIO	TOTAL_TIME	TOTAL_COUNT	P999	P99	P90	P80
tidb_query  expand	1	23223.86	459625	63.54	45.55	0.1	0.06	
tidb_get_token  expand	0.000001	0.03	458888	0.00004	0.000001	0.000001	0.000001	
tidb_parse  expand	0.00001	0.29	4603	0.0006	0.0004	0.0002	0.0001	
tidb_compile  expand	0.0008	17.82	463488	0.001	0.0009	0.0003	0.0002	
tidb_execute  expand	1	23237.39	463491	90.81	0.35	0.1	0.08	
tidb_distsql_execution  expand	0.0002	4.53	1064	0.13	0.1	0.02	0.01	
tidb_cop  expand	0.0005	12.39	1075	2.03	1.84	0.02	0.01	
Transaction  expand	1	23158.73	460023	8.16	7.91	5.38	0.06	
tidb_transaction_local_latch_wait  expand	0	0	0					
tidb_kv_backoff  expand	0.00003	0.58	291	0.002	0.002	0.002	0.002	
KV request  expand	1.65	38325.58	2300949	2.03	1.84	0.04	0.03	
Slow query  expand	0.01	323.65	280	65.5	65.21	62.26	58.98	
tidb_slow_query_cop_process  expand	0	0	280	0.001	0.001	0.0009	0.0008	
tidb_slow_query_cop_wait  expand	0	0	280	0.001	0.001	0.0009	0.0008	
DDL job  expand	0	0	0					
tidb_ddl_worker  expand	0	0	0					
tidb_ddl_update_self_version  expand	0	0	0					
tidb_owner_handle_syncer  expand	0	0	0					
Batch add index  expand								
tidb_ddl_deploy_syncer  expand	0	0	0					
Schema load  expand	0.000008	0.19	27	0.06	0.06	0.06	0.05	
TiDB meta operation  expand	0.0001	3.21	1200	0.13	0.1	0.01	0.005	
TiDB auto id request  expand	0	0	0					
TiDB auto analyze  expand	0.003	78.35	1	81.88	81.51	77.82	73.73	
TiDB GC  expand	0.000000001	0.00002	3	1	0.99	0.9	0.8	
tidb_gc_push_task  expand	0	0	0					
tidb_batch_client_unavailable  expand	0	0	0					
tidb_batch_client_wait  expand	0	0	0					
pd_tso_rpc  expand	0.005	123.45	108540	0.02	0.008	0.003	0.002	
pd_tso_wait  expand	0.07	1662.29	920545	0.03	0.01	0.004	0.003	
PD client cmd  expand	0.15	3370.87	2761712	0.03	0.01	0.004	0.003	
pd_client_request_rpc  expand	0.005	123.45	108540	0.02	0.008	0.003	0.002	
pd_grpc_completed_commands  expand	0.00005	1.24	5255	0.01	0.01	0.006	0.005	
pd_operator_finish  expand								
pd_operator_step_finish  expand								
Etcd transactions  expand	0.000008	0.17	207	0.008	0.008	0.006	0.002	
pd_region_heartbeat  expand	0.00006	1.33	195	0	0	0	0	
etcd_wal_fsync  expand	0.00001	0.33	288	0.02	0.01	0.006	0.004	
nd peer round trip  expand								

Figure 407: Time Consume report

The fields in columns of the table above are described as follows:

- **METRIC_NAME**: The name of the monitoring metric.
- **Label**: The label information for the monitoring metric. Click **expand** to view more detailed monitoring information of each label for a metric.
- **TIME_RATIO**: The ratio of the total time consumed by this monitoring metric to the total time of the monitoring row where **TIME_RATIO** is 1. For example, the total

consumed time by `kv_request` is 1.65 (namely, $38325.58/23223.86$) times that of `tidb_query`. Because KV requests are executed concurrently, the total time of all KV requests might exceed the total query (`tidb_query`) execution time.

- `TOTAL_TIME`: The total time consumed by this monitoring metric.
- `TOTAL_COUNT`: The total number of times this monitoring metric is executed.
- `P999`: The maximum P999 time of this monitoring metric.
- `P99`: The maximum P99 time of this monitoring metric.
- `P90`: The maximum P90 time of this monitoring metric.
- `P80`: The maximum P80 time of this monitoring metric.

The following image shows the relationship of time consumption of the related modules in the monitoring metrics above.

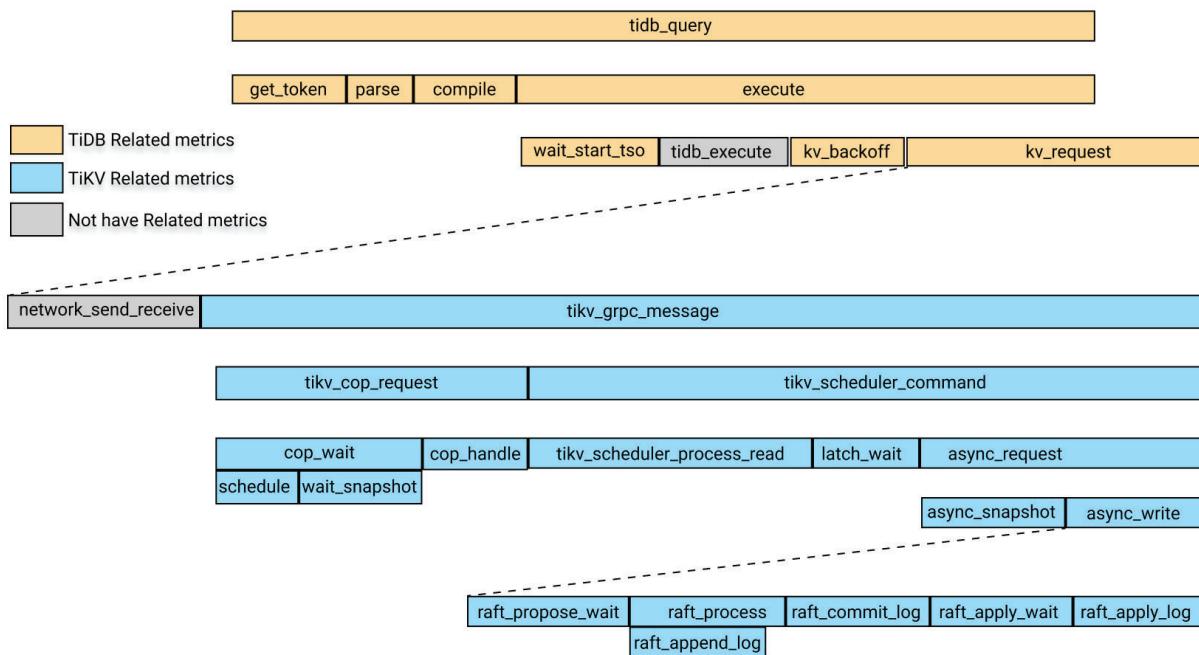


Figure 408: Time-consumption relationship of each module

In the image above, yellow boxes are the TiDB-related monitoring metrics. Blue boxes are TiKV-related monitoring metrics, and gray boxes temporarily do not correspond to specific monitoring metrics.

In the image above, the time consumption of `tidb_query` includes the following four parts:

- `get_token`
- `parse`
- `compile`

- `execute`

The `execute` time includes the following parts:

- `wait_start_tso`
- The execution time at the TiDB layer, which is currently not monitored
- KV request time
- `KV_backoff` time, which is the time for backoff after the KV request fails

Among the parts above, the KV request time includes the following parts:

- The time consumed by the network sending and receiving of requests. Currently, there is no monitoring metric for this item. You can subtract the time of `tikv_grpc_message` from the KV request time to roughly estimate this item.
- `tikv_grpc_message` time consumption.

Among the parts above, `tikv_grpc_message` time consumption includes the following parts:

- Coprocessor request time consumption, which refers to processing the COP type requests. This time consumption includes the following parts:
 - `tikv_cop_wait`: The time consumed by request queue.
 - Coprocessor handle: The time consumed to process Coprocessor requests.
- `tikv_scheduler_command` time consumption, which includes the following parts:
 - `tikv_scheduler_processing_read`: The time consumed to process read requests.
 - The time consumed to get snapshot in `tikv_storage_async_request` (snapshot is the label for this monitoring metric).
 - Time consumed to process write requests. This time consumption includes the following parts:
 - * `tikv_scheduler_latch_wait`: The time consumed to wait for latch.
 - * The time consumption of writes in `tikv_storage_async_request` (write is the label for this monitoring metric).

Among the above metrics, The time consumption of writes in `tikv_storage_async_request` refers to the time consumption of writing Raft KVs, including the following parts:

- `tikv_raft_propose_wait`
- `tikv_raft_process`, which mainly includes `tikv_raft_append_log`
- `tikv_raft_commit_log`
- `tikv_raft_apply_wait`

- `tikv_raft_apply_log`

You can use `TOTAL_TIME`, the P999 time, and the P99 time to determine which modules consume longer time according to the relationship between the time consumptions described above, and then look at the related monitoring metrics.

Note:

Because the Raft KV writes might be processed in one batch, using `TOTAL_TIME` to measure the time consumed by each module is inapplicable to monitoring metrics related to Raft KV writes, specifically, `tikv_raft_process`, `tikv_raft_append_log`, `tikv_raft_commit_log`, `tikv_raft_apply_wait`, and `tikv_raft_apply_log`. In this situation, it is more reasonable to compare the time consumption of each module with the time of P999 and P99.

The reason is that if there are 10 asynchronous write requests, Raft KV internally pack 10 requests into a batch execution, and the execution time is 1 second. Therefore, the execution time of each request is 1 second, and the total time of 10 requests is 10 seconds, but the total time for Raft KV processing is 1 second. If you use `TOTAL_TIME` to measure the consumed time, you might not understand where the remaining 9 seconds are spent. You can also see the difference between the monitoring metric of Raft KV and other previous monitoring metrics from the total number of requests (`TOTAL_COUNT`).

Errors Occurred in Each Component

The `Errors Occurred in Each Component` table shows the total number of errors in TiDB and TiKV, such as the failure to write binlog, `tikv server is busy`, `TiKV channel full`, `tikv write stall`. You can see the row comments for the specific meaning of each error.

METRIC_NAME	LABEL	TOTAL_COUNT
tidb_binlog_error_total_count ⓘ		0
tidb_handshake_error_total_count ⓘ		0
tidb_transaction_retry_error_total_count ⓘ		
tidb_kv_region_error_total_count ⓘ expand		903
tidb_schema_lease_error_total_count ⓘ		0
tikv_grpc_error_total_count ⓘ		0
tikv_critical_error_total_count ⓘ		
tikv_scheduler_is_busy_total_count ⓘ		
tikv_channel_full_total_count ⓘ		
tikv_coprocessor_request_error_total_count ⓘ expand		1
tikv_engine_write_stall ⓘ		0
tikv_server_report_failures_total_count ⓘ expand		1396
tikv_storage_async_request_error ⓘ expand		1176
tikv_lock_manager_detect_error_total_count ⓘ		
tikv_backup_errors_total_count ⓘ		
node_network_in_errors_total_count ⓘ		0
node_network_out_errors_total_count ⓘ		0

Figure 409: Errors Occurred in Each Component report

Specific TiDB/PD/TiKV monitoring information

This part includes more specific monitoring information of TiDB, PD, or TiKV.

TiDB-related monitoring information

Time Consumed by TiDB Component

This table shows the time consumed by each TiDB module and the ratio of each time consumption, which is similar to the `time consume` table in the overview, but the label information of this table are more detailed.

TiDB Server Connections

This table shows the number of client connections for each TiDB instance.

TiDB Transaction

This table shows transaction-related monitoring metrics.

METRIC_NAME	LABEL	TOTAL_VALUE	TOTAL_COUNT	P999	P99	P90	P80
tidb_transaction_retry_num ⓘ		0	0	0	0	0	0
tidb_transaction_statement_num ⓘ expand		6854588	6852371	4	4	3	2
tidb_txn_region_num ⓘ expand		1070770	706887	3	2	2	2
tidb_txn_kv_write_num ⓘ expand		513386	181312	234	2	2	2
tidb_txn_kv_write_size ⓘ expand		266.772 MB	181296	116.913 KB	1.996 KB	1.905 KB	1.805 KB
tidb_load_safepoint_total_num ⓘ expand		16					
tidb_lock_resolver_total_num ⓘ expand		420514					

Figure 410: Transaction report

- **TOTAL_VALUE**: The sum of all values (SUM) during the report time range.
- **TOTAL_COUNT**: The total number of occurrences of this monitoring metric.
- **P999**: The maximum P999 value of this monitoring metric.
- **P99**: The maximum P99 value of this monitoring metric.
- **P90**: The maximum P90 value of this monitoring metric.
- **P80**: The maximum P80 value of this monitoring metric.

Example:

In the table above, within the report time range, `tidb_txn_kv_write_size`: a total of about 181,296 transactions of KV writes, and the total KV write size is 266.772 MB, of which the maximum P999, P99, P90, P80 values for a single transaction of KV writes are 116.913 KB, 1.996 KB, 1.905 KB, and 1.805 KB.

DDL Owner

MIN_TIME	DDL OWNER
2020-05-21 14:40:00	10.0.1.13:10080

Figure 411: TiDB DDL Owner Report

The table above shows that from 2020-05-21 14:40:00, the cluster's DDL OWNER is at the 10.0.1.13:10080 node. If the owner changes, multiple rows of data exist in the table above, where the `Min_Time` column indicates the minimum time of the corresponding known owner.

Note:

If the owner information is empty, it does not mean that no owner exists in this period of time. Because in this situation, the DDL owner is determined based on the monitoring information of `ddl_worker`, it might be that `ddl_worker` → has not done any DDL job in this period of time, causing the owner information to be empty.

Other monitoring tables in TiDB are as follows:

- Statistics Info: Shows related monitoring metrics of TiDB statistical information.
- Top 10 Slow Query: Shows the Top 10 slow query information in the report time range.
- Top 10 Slow Query Group By Digest: Shows the Top 10 slow query information in the report time range, which is aggregated according to the SQL fingerprint.

- Slow Query With Diff Plan: The SQL statement whose execution plan changes within the report time range.

PD related monitoring information

The tables related to the monitoring information of PD modules are as follows:

- **Time Consumed by PD Component:** The time consumed by the monitoring metrics of related modules in PD.
- **Balance Leader/Region:** The monitoring information of `balance-region` and `balance leader` occurred in the cluster within the report time range, such as the number of leaders that are scheduled out from `tikv_note_1` or the number of leaders that are scheduled in.
- **Cluster Status:** The cluster status information, including total number of TiKV nodes, total cluster storage capacity, the number of Regions, and the number of offline TiKV nodes.
- **Store Status:** Record the status information of each TiKV node, including Region score, leader score, and the number of Regions/leaders.
- **Etcd Status:** etcd related information in PD.

TiKV related monitoring information

The tables related to the monitoring information of TiKV modules are as follows:

- **Time Consumed by TiKV Component:** The time consumed by related modules in TiKV.
- **Time Consumed by RocksDB:** The time consumed by RocksDB in TiKV.
- **TiKV Error:** The error information related to each module in TiKV.
- **TiKV Engine Size:** The size of stored data of column families on each node in TiKV.
- **Coprocessor Info:** Monitoring information related to the Coprocessor module in TiKV.
- **Raft Info:** Monitoring information of the Raft module in TiKV.
- **Snapshot Info:** Snapshot related monitoring information in TiKV.
- **GC Info:** Garbage Collection (GC) related monitoring information in TiKV.
- **Cache Hit:** The hit rate information of each cache of RocksDB in TiKV.

Configuration information

In the configuration information, the configuration values of some modules are shown within the report time range. But the historical values of some other configurations of these modules cannot be obtained, so the shown values of these configurations are the current (when the report is generated) values .

Within the report time range, the following tables include items whose values are configured at the start time of the report time range:

- **Scheduler Initial Config:** The initial value of PD scheduling-related configuration at the report's start time.
- **TiDB GC Initial Config:** The initial value of TiDB GC related-configuration at the report's start time
- **TiKV RocksDB Initial Config:** The initial value of TiKV RocksDB-related configuration at the report's start time
- **TiKV RaftStore Initial Config:** The initial value of TiKV RaftStore-related configuration at the report's start time

Within the report time range, if some configurations have been modified, the following tables include records of some configurations that have been modified:

- Scheduler Config Change History
- TiDB GC Config Change History
- TiKV RocksDB Config Change History
- TiKV RaftStore Config Change History

Example:

APPROXIMATE_CHANGE_TIME	CONFIG_ITEM	VALUE
2020-05-22T20:00:00+08:00	leader-schedule-limit	4
2020-05-22T20:07:00+08:00	leader-schedule-limit	8

Figure 412: Scheduler Config Change History report

The table above shows that the `leader-schedule-limit` configuration parameter has been modified within the report time range:

- 2020-05-22T20:00:00+08:00: At the start time of the report, the configuration value of `leader-schedule-limit` is 4, which does not mean that the configuration has been modified, but that at the start time in the report time range, its configuration value is 4.
- 2020-05-22T20:07:00+08:00: The `leader-schedule-limit` configuration value is 8 ↵ , which indicates that the value of this configuration has been modified around 2020-05-22T20:07:00+08:00.

The following tables show the current configuration of TiDB, PD, and TiKV at the time when the report is generated:

- **TiDB's Current Config**
- **PD's Current Config**
- **TiKV's Current Config**

Comparison report

You can generate a comparison report for two time ranges. The report content is the same as the report for a single time range, except that a comparison column is added to show the difference between the two time ranges. The following sections introduce some unique tables in the comparison report and how to view the comparison report.

First, the `Compare Report Time Range` report in the basic information shows the two time ranges for comparison:

Compare Report Time Range			
T1.START_TIME	T1.END_TIME	T2.START_TIME	T2.END_TIME
2020-05-21 06:30:00	2020-05-21 06:35:00	2020-05-21 06:40:00	2020-05-21 06:45:00

Figure 413: Compare Report Time Range report

In the table above, `t1` is the normal time range, or the reference time range. `t2` is the abnormal time range.

Tables related to slow queries are shown as follows:

- `Slow Queries In Time Range t2`: Shows slow queries that only appear in `t2` but not during `t1`.
- `Top 10 slow query in time range t1`: The Top 10 slow queries during `t1`.
- `Top 10 slow query in time range t2`: The Top 10 slow queries during `t2`.

DIFF_RATIO introduction

This section introduces DIFF_RATIO using the `Instance CPU Usage` table as an example.

Instance CPU Usage										
INSTANCE	JOB	t1.AVG	t1.MAX	t1.MIN	t2.AVG	t2.MAX	t2.MIN	AVG_DIFF_RATIO	MAX_DIFF_RATIO	MIN_DIFF_RATIO
172.16.5.40:10089	tidb	410%	410%	410%	1240%	1240%	1240%	2.02	2.02	2.02
172.16.5.40:20151	tikv	221%	221%	221%	617%	617%	617%	1.79	1.79	1.79
172.16.5.40:20379	pd	82%	82%	82%	78%	78%	78%	-0.05	-0.05	-0.05

Figure 414: Compare Instance CPU Usage report

- `t1.AVG`, `t1.MAX`, `t1.Min` are the average value, maximum value, and minimum value of CPU usage in the `t1`.
- `t2.AVG`, `t2.MAX`, and `t2.Min` are the average value, maximum value, and minimum value of CPU usage during `t2`.
- `AVG_DIFF_RATIO` is DIFF_RATIO of the average values during `t1` and `t2`.
- `MAX_DIFF_RATIO` is DIFF_RATIO of the maximum values during `t1` and `t2`.
- `MIN_DIFF_RATIO` is DIFF_RATIO of the minimum values during `t1` and `t2`.

DIFF_RATIO: Indicates the difference value between the two time ranges. It has the following values:

- If the monitoring metric has a value only within t_2 and has no value within t_1 , the value of DIFF_RATIO is 1.
- If the monitoring metric has a value only within t_1 , and has no value within t_2 time range, the value of DIFF_RATIO is -1.
- If the value of t_2 is greater than that of t_1 , then $\text{DIFF_RATIO} = (\frac{t_2.\text{value}}{t_1.\text{value}}) - 1$
- If the value of t_2 is smaller than that of t_1 , then $\text{DIFF_RATIO} = 1 - (\frac{t_1.\text{value}}{t_2.\text{value}})$

For example, in the table above, the average CPU usage of the `tidb` node in t_2 is 2.02 times higher than that in t_1 , which is $2.02 = \frac{1240}{410} - 1$.

Maximum Different Item table

The **Maximum Different Item** table compares the monitoring metrics of two time ranges, and sorts them according to the difference of the monitoring metrics. Using this table, you can quickly find out which monitoring metric has the biggest difference in the two time ranges. See the following example:

Maximum Different Item

The maximum different metrics between two time ranges.

TABLE	METRIC_NAME	LABEL	MAX_DIFF	t1.VALUE	t2.VALUE	VALUE_TYPE
TiKV, coprocessor_info Expand	TiKV Coprocessor scan	172.16.5.40:20151,select,get,lock	28916.85	61.33	1773532	TOTAL_VALUE
TiDB, statistics_info	store_query_feedback_total_count	172.16.5.40:10089,ok	7219.00		7219	TOTAL_COUNT
overview, total_time_consume	tidb_parse	general	1201.00		1201	TOTAL_COUNT
TiDB, tidb_time_consume	tidb_slow_query_cop_wait	172.16.5.40:10089	800.00		800	TOTAL_COUNT
overview, total_time_consume	tidb_slow_query_cop_process	172.16.5.40:10089	800.00		800	TOTAL_COUNT
overview, total_time_consume	Slow query	172.16.5.40:10089	799.00	1	800	TOTAL_COUNT
TiKV, coprocessor_info	TiKV Coprocessor response	172.16.5.40:20151	534.43	15.822 MB	8.273 GB	TOTAL_VALUE
overview, total_time_consume	tidb_distsql_execution	dag	470.06	0.68	320.32	TOTAL_TIME
load, tikv_thread_cpu_usage	unified read pool	172.16.5.40:20151	412.29	0.07%	28.93%	MIN
TiKV, coprocessor_info Expand	TiKV Coprocessor scan keys	172.16.5.40:20151,select	198.39	1067055.38	212755381.33	TOTAL_VALUE
overview, total_time_consume Expand	tidb_ddl_worker	drop view	139.00		139	TOTAL_COUNT
overview, total_error Expand	tikv_storage_async_request_error	snapshot	89.00		89	TOTAL_COUNT
overview, total_time_consume Expand	Coprocessor handling request	index	78.99	0.67	53.59	TOTAL_TIME
overview, total_time_consume	tidb_cop	172.16.5.40:10089	75.11	5.93	451.31	TOTAL_TIME
overview, total_time_consume	KV request	Cop	75.10	5.93	451.26	TOTAL_TIME
overview, total_time_consume Expand	Coprocessor request	select	73.12	4.73	350.6	TOTAL_TIME
PD, pd_time_consume Expand	pd_region_heartbeat	172.16.5.40:20150,1	-73.00	73		TOTAL_COUNT
overview, total_time_consume Expand	tikv_grpc_message	coprocessor	71.75	5.56	404.49	TOTAL_TIME
TiDB, tidb_time_consume	tidb_ddl_update_self_version	172.16.5.40:10089,ok	71.00		71	TOTAL_COUNT
overview, total_time_consume Expand	TiDB meta operation	update_ddl_job	71.00		71	TOTAL_COUNT
overview, total_time_consume Expand	tidb_owner_handle_syncer	update_global_version	71.00		71	TOTAL_COUNT
TiDB, tidb_time_consume Expand	tidb_query	172.16.5.40:10089,internal	-62.84	15.96	0.25	P999
TiKV, scheduler_info Expand	Scheduler stage	172.16.5.40:20151,scan_lock,snapshot_ok	55.45		55.45	TOTAL_VALUE
overview, total_time_consume Expand	tikv_grpc_message	kv_scan_lock	55.00		55	TOTAL_COUNT
TiKV, tikv_time_consume Expand	tikv_scheduler_command	172.16.5.40:20151,scan_lock	55.00		55	TOTAL_COUNT
overview, total_time_consume Expand	PD client cmd	get_region	44.00		44	TOTAL_COUNT
overview, total_time_consume Expand	tikv_cop_wait	select	39.40	171	6908	TOTAL_COUNT
TiDB, tidb_time_consume	Schema load	172.16.5.40:10089	37.50	0.02	0.77	TOTAL_TIME
overview, total_time_consume	Schema load	172.16.5.40:10089	30.25	0.008	0.25	P999

Figure 415: Maximum Different Item table

- **Table:** Indicates this monitoring metric comes from which table in the comparison report. For example, TiKV, coprocessor_info indicates the coprocessor_info table in the TiKV component.
- **METRIC_NAME:** The monitoring metric name. Click expand to view the comparison of different labels of metrics.
- **LABEL:** The label corresponding to the monitoring metric. For example, the monitoring metric of TiKV Coprocessor scan has 2 labels, namely instance, req, tag, sql_type ↗, which are the TiKV address, request type, operation type and operation column family.
- **MAX_DIFF:** Difference value, which is the DIFF_RATIO calculation of t1.VALUE and

t2.VALUE.

From the table above, you can see the **t2** time range has much more Coprocessor requests than the **t1** time range, and the SQL parsing time of TiDB in **t2** is much longer.

11.6.1.10.3 Locate Problems Using Diagnostic Report of TiDB Dashboard

This document introduces how to locate problems using diagnostic report of TiDB Dashboard.

Comparison diagnostics

This section demonstrates how to use the comparison diagnostic feature to diagnose QPS jitter or latency increase caused by large queries or writes.

Example 1

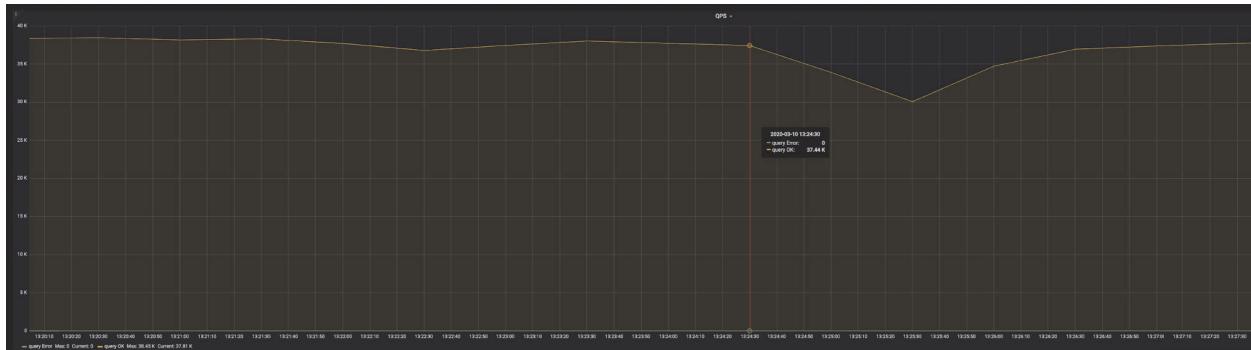


Figure 416: QPS example

The result of a go-ycsb pressure test is shown in the image above. You can see that at 2020-03-10 13:24:30, QPS suddenly began to decrease. After 3 minutes, QPS began to return to normal. You can use diagnostic report of TiDB Dashboard to find out the cause.

Generate a report that compares the system in the following two time ranges:

T1: 2020-03-10 13:21:00 to 2020-03-10 13:23:00. In this range, the system is normal, which is called a reference range.

T2: 2020-03-10 13:24:30 to 2020-03-10 13:27:30. In this range, QPS began to decrease.

Because the impact range of jitter is 3 minutes, the two time ranges above are both 3 minutes. Because some monitored average values are used for comparison during diagnostics, too long a range will cause the difference of average values to be insignificant, and then the problem cannot be accurately located.

After the report is generated, you can view this report on the **Compare Diagnose** page.

Compare Diagnose

Automatically diagnose the cluster problem by compare with the refer time.

RULE	DETAIL
big-query	may have big query in diagnose time range
fold	
--	tidb_qps,172.16.5.40:10089: ↓ 0.93 (34927.00 / 37658.00)
--	tidb_query_duration,172.16.5.40:10089: ↑ 1.54 (0.25 / 0.16)
--	tidb_cop_duration,172.16.5.40:10089: ↑ 2.48 (0.16 / 0.06)
--	tidb_kv_write_num,172.16.5.40:10089: ↑ 7.61 (1766.40 / 232.25)
--	tikv_cop_scan_keys_total_num,172.16.5.40:22151: ↑ 136446.22 (98242004.00 / 720.01)
--	tikv_cop_scan_keys_total_num,172.16.5.40:23151: ↑ 65079325.09 (65079325.09 / 0.00)
--	tikv_cop_scan_keys_total_num,172.16.5.40:20151: ↑ 46976.83 (101469210.67 / 2159.98)
--	pd_operator_step_finish_total_count,TransferLeader: ↑ 2.45 (36.00 / 14.67)
--	try to check the slow query only appear in diagnose time range with sql: SELECT * FROM (SELECT count(*), min(time), sum(query_time) AS sum_query_time, sum(Process_time) AS sum_process_time, sum(Wait_time) AS sum_wait_time, sum(Commit_time), sum(Request_count), sum(process_keys), sum(Write_keys), max(Cop_proc_max), min(query),min(prev_stmt), digest FROM information_schema.CLUSTER_SLOW_QUERY WHERE time >= '2020-03-10 13:24:30' AND time < '2020-03-10 13:27:30' AND Is_internal = false GROUP BY digest) AS t1 WHERE t1.digest NOT IN (SELECT digest FROM information_schema.CLUSTER_SLOW_QUERY WHERE time >= '2020-03-10 13:21:00' AND time < '2020-03-10 13:24:00' GROUP BY digest) ORDER BY t1.sum_query_time DESC limit 10;

Figure 417: Comparison diagnostics

The diagnostic results above show that big queries might exist during the diagnostic time. Each **DETAIL** in the report above is described as follows:

- **tidb_qps**: QPS decreased by 0.93 times.
- **tidb_query_duration**: P999 query latency increased by 1.54 times.
- **tidb_cop_duration**: The processing latency of P999 coprocessor requests increased by 2.48 times.
- **tidb_kv_write_num**: The number written KVs in the P999 TiDB transaction increased by 7.61 times.
- **tikv_cop_scan_keys_total_nun**: The number of keys/values scanned by the TiKV Coprocessor has greatly improved on 3 TiKV instances.
- In **pd_operator_step_finish_total_count**, the number of transferred leaders increases by 2.45 times, which means that the scheduling in the abnormal time range is higher than that in the normal time range.
- The report indicates that there might be slow queries and indicates that you can use SQL statements to query slow queries. The execution result of the SQL statements are as follows:

```
SELECT * FROM (SELECT count(*), min(time), sum(query_time) AS
    → sum_query_time, sum(Process_time) AS sum_process_time, sum(Wait_time)
    → AS sum_wait_time, sum(Commit_time), sum(Request_count), sum(
    → process_keys), sum(Write_keys), max(Cop_proc_max), min(query),min(
    → prev_stmt), digest FROM information_schema.CLUSTER_SLOW_QUERY WHERE
    → time >= '2020-03-10 13:24:30' AND time < '2020-03-10 13:27:30' AND
    → Is_internal = false GROUP BY digest) AS t1 WHERE t1.digest NOT IN (
    → SELECT digest FROM information_schema.CLUSTER_SLOW_QUERY WHERE time
```

```

    ↵ >= '2020-03-10 13:21:00' AND time < '2020-03-10 13:24:00' GROUP BY
    ↵ digest) ORDER BY t1.sum_query_time DESC limit 10\G
*****[ 1. row ]*****
count(*) | 196
min(time) | 2020-03-10 13:24:30.204326
sum_query_time | 46.878509117
sum_process_time | 265.924
sum_wait_time | 8.308
sum(Commit_time) | 0.926820886
sum(Request_count) | 6035
sum(process_keys) | 201453000
sum(Write_keys) | 274500
max(Cop_proc_max) | 0.263
min(query) | delete from test.tcs2 limit 5000;
min(prev_stmt) |
digest | 24
    ↵ bd6d8a9b238086c9b8c3d240ad4ef32f79ce94cf5a468c0b8fe1eb5f8d03df

```

From the result above, you can see that from 13:24:30, there is a large write of batch deletion, which has been executed 196 times in total, each time deleting 5,000 rows of data, in a total duration of 46.8 seconds.

Example 2

If a large query has not been executed, the query is not recorded in the slow log. In this situation, this large query can still be diagnosed. See the following example:



Figure 418: QPS results

The result of another go-ycsb pressure test is shown in the image above. You can see that at 2020-03-08 01:46:30, QPS suddenly began to drop and did not recover.

Generate a report that compares the system in the following two time ranges:

T1: 2020-03-08 01:36:00 to 2020-03-08 01:41:00. In this range, the system is normal, which is called a reference range.

T2: 2020-03-08 01:46:30 to 2020-03-08 01:51:30. In this range, QPS began to decrease.

After the report is generated, you can view this report on the **Compare Diagnose** page.

Compare Diagnose

Automatically diagnose the cluster problem by compare with the refer time.

RULE	DETAIL
big-query fold	may have big query in diagnose time range
--	tidb_qps,172.16.5.40:10089: ↓ 0.85 (31881.00 / 37674.00)
--	tidb_query_duration,172.16.5.40:10089: ↑ 1.35 (0.06 / 0.04)
--	tidb_cop_duration,172.16.5.40:10089: ↑ 3.78 (0.08 / 0.02)
--	tidb_kv_write_num,172.16.5.40:10089: ↑ 511.74 (511.74 / 0.00)
--	tikv_cop_scan_keys_total_num,172.16.5.40:20151: ↑ 1277.21 (6270285.33 / 4909.36)
--	try to check the slow query only appear in diagnose time range with sql: SELECT * FROM information_schema.cluster_log WHERE type='tidb' AND time >= '2020-03-08 01:46:30' AND time < '2020-03-08 01:51:30' AND level = 'warn' AND message LIKE '%expensive_query%'

Figure 419: Comparison diagnostics

The diagnostic result is similar to that of example 1. The last row of the image above indicates that there might be slow queries and indicate that you can use SQL statements to query the expensive queries in the TiDB log. The execution result of the SQL statements are as follows.

```
> SELECT * FROM information_schema.cluster_log WHERE type='tidb' AND time
  ↪ >= '2020-03-08 01:46:30' AND time < '2020-03-08 01:51:30' AND level =
  ↪ 'warn' AND message LIKE '%expensive_query%' \G
TIME    | 2020/03/08 01:47:35.846
TYPE    | tidb
INSTANCE | 172.16.5.40:4009
LEVEL   | WARN
MESSAGE | [expensivequery.go:167] [expensive_query] [cost_time=60.085949605s
  ↪ ] [process_time=2.52s] [wait_time=2.52s] [request_count=9] [
  ↪ total_keys=996009] [process_keys=996000] [num_cop_tasks=9] [
  ↪ process_avg_time=0.28s] [process_p90_time=0.344s] [process_max_time
  ↪ =0.344s] [process_max_addr=172.16.5.40:20150] [wait_avg_time
  ↪ =0.000777777s] [wait_p90_time=0.003s] [wait_max_time=0.003s] [
  ↪ wait_max_addr=172.16.5.40:20150] [stats=t_wide:pseudo] [conn_id
  ↪ =19717] [user=root] [database=test] [table_ids="[80,80]" [
  ↪ txn_start_ts=415132076148785201] [mem_max="23583169 Bytes
  ↪ (22.490662574768066 MB)"] [sql="select count(*) from t_wide as t1
  ↪ join t_wide as t2 where t1.c0>t2.c1 and t1.c2>0"]]
```

The query result above shows that on this 172.16.5.40:4009 TiDB instance, at 2020/03/08 01:47:35.846 there is an expensive query that has been executed for 60

seconds, but the execution has not yet been finished. This query is a join of Cartesian products.

Locate problems using comparison diagnostic report

Because the diagnostics might be wrong, using a comparison report might help DBAs locate problems more quickly. See the following example.

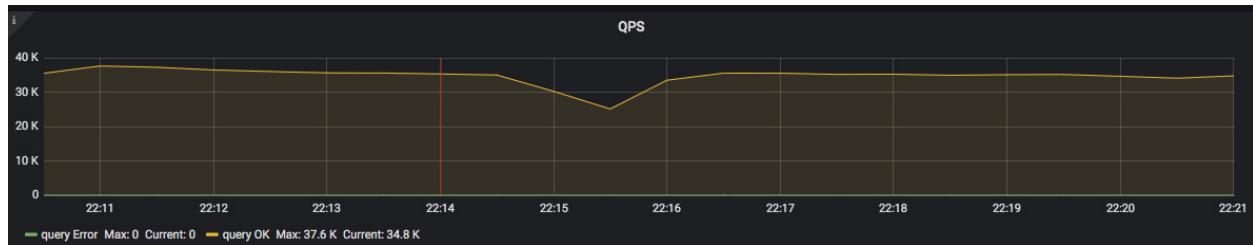


Figure 420: QPS results

The result of a go-ycsb pressure test is shown in the image above. You can see that at 2020-05-22 22:14:00, QPS suddenly began to decrease. After 3 minutes, QPS began to return to normal. You can use the comparison diagnostic report of TiDB Dashboard to find out the cause.

Generate a report that compares the system in the following two time ranges:

T1: 2020-05-22 22:11:00 to 2020-05-22 22:14:00. In this range, the system is normal, which is called a reference range.

T2: 2020-05-22 22:14:00 2020-05-22 22:17:00. In this range, QPS began to decrease.

After generating the comparison report, check the **Max diff item** report. This report compares the monitoring items of the two time ranges above and sorts them according to the difference of the monitoring items. The result of this table is as follows:

Maximum Different Item

The maximum different metrics between two time ranges.

TABLE	METRIC_NAME	LABEL	MAX_DIFF	t1.VALUE	t2.VALUE	VALUE_TYPE
TiDB, transaction	Transaction KV write count	172.16.5.40:10089	-21616.00	43234	2	P999
TiKV, coprocessor_info Expand	TiKV Coprocessor scan	172.16.5.40:20151,select,get,lock	10499.24	64	672015.48	TOTAL_VALUE
TiKV, coprocessor_info Expand	TiKV Coprocessor scan keys	172.16.5.40:20151,select	4264.45	15431.85	65823838.67	TOTAL_VALUE
TiDB, transaction	Transaction KV write size	172.16.5.40:10089	-2702.69	5.278 MB	1.999 KB	P999
overview, total_time_consume	Coprocessor handling request	select	2132.50	0.06	128.01	TOTAL_TIME
TiKV, coprocessor_info	TiKV Coprocessor response	172.16.5.40:20151	1535.00	1.792 MB	2.688 GB	TOTAL_VALUE
TiDB, statistics_info	store_query_feedback_total_count	172.16.5.40:10089,ok	1447.00		1447	TOTAL_COUNT
TiKV, scheduler_info Expand	tikv_scheduler_keys_written	172.16.5.40:20151,commit	-860.00	861	1	P99
TiKV, tikv_time_consume	Coprocessor request	172.16.5.40:20151,select	853.87	0.15	128.23	TOTAL_TIME
load, tikv_thread_cpu_usage	unified read pool	172.16.5.40:20151	667.27	0.11%	73.51%	AVG
TiKV, tikv_time_consume	Coprocessor handling request	172.16.5.40:20151,index	444.00	0.002	0.89	P999
TiKV, tikv_time_consume Expand	tikv_grpc_message	172.16.5.40:20151,coprocessor	419.97	0.33	138.92	TOTAL_TIME
load, node_load_info	tikv_disk_read_bytes	172.16.5.40:19110,sda	-273.41	0.267 KB	0.000 KB	MAX
overview, total_time_consume Expand	KV request	Cop	244.29	0.66	161.89	TOTAL_TIME

Figure 421: Comparison results

From the result above, you can see that the Coprocessor requests in T2 are much more than those in T1. It might be that some large queries appear in T2 that bring more load.

In fact, during the entire time range from T1 to T2, the `go-ycsb` pressure test is running. Then 20 `tpch` queries are running during T2. So it is the `tpch` queries that cause many Coprocessor requests.

If such a large query whose execution time exceeds the threshold of slow log is recorded in the slow log. You can check the `Slow Queries In Time Range t2` report to see whether there is any slow query. However, some queries in T1 might become slow queries in T2, because in T2, other loads cause their executions to slow down.

11.6.1.11 TiDB Dashboard Log Search Page

On the log search page of TiDB Dashboard, you can search logs of all nodes, preview the search result and download logs.

11.6.1.11.1 Access the page

After logging into TiDB Dashboard, you can click **Search Logs** to enter this log search homepage.

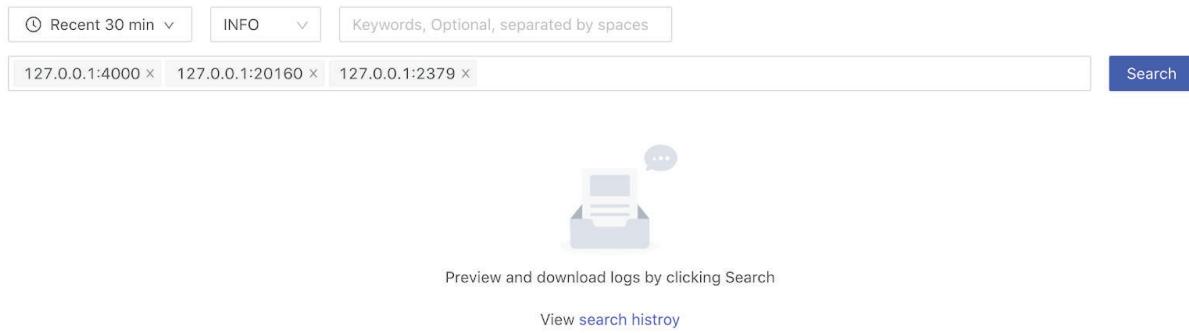


Figure 422: Log Search Page

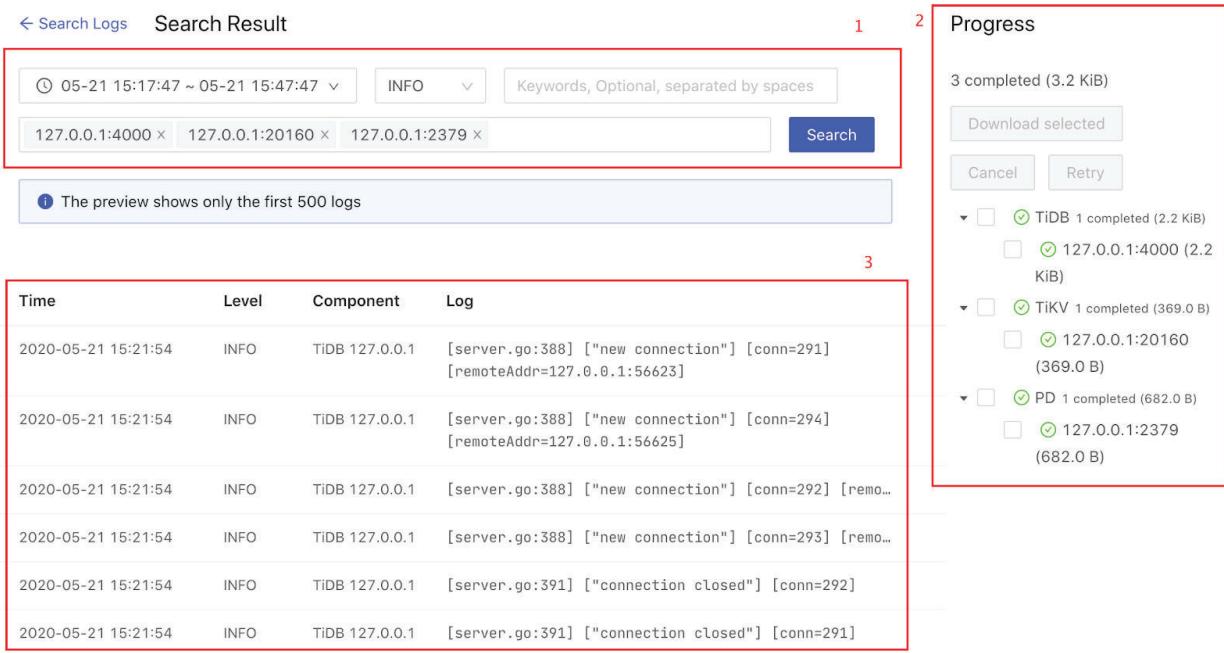
This page provides the following search parameters:

- Time range: Specifies the time range of logs to search. The default value is the recent 30 minutes.
- Log level: Specifies the minimum log level. All logs above this log level are searched. The default value is the `INFO`.
- Keywords: The parameter is optional and its value can be any legal string. Multiple keywords are separated by a space. Regular expressions are supported (case-insensitive).
- Components: Selects the cluster components to search, which are multi-select and non-empty. By default, all components are selected.

After clicking the **Search** button, you enter the detail page of the search results.

11.6.1.11.2 Page of search result

The following image shows the page of the search results.



The screenshot shows the 'Search Result' page with three numbered areas:

- Area 1 (Parameter options):** Contains search filters for time range (05-21 15:17:47 ~ 05-21 15:47:47), log level (INFO), and keywords. It also shows the selected log addresses: 127.0.0.1:4000, 127.0.0.1:20160, and 127.0.0.1:2379.
- Area 2 (Progress):** Shows the progress of log downloads for three components: TiDB, TiKV, and PD. All are marked as completed with file sizes: 3.2 KiB for TiDB, 2.2 KiB for 127.0.0.1:4000, 369.0 B for 127.0.0.1:20160, and 682.0 B for 127.0.0.1:2379.
- Area 3 (Search results):** A table listing log entries. The columns are Time, Level, Component, and Log. The data shows multiple log entries from 2020-05-21 15:21:54 for component TiDB at level INFO, detailing new connections and connection closed events.

Time	Level	Component	Log
2020-05-21 15:21:54	INFO	TiDB 127.0.0.1	[server.go:388] ["new connection"] [conn=291] [remoteAddr=127.0.0.1:56623]
2020-05-21 15:21:54	INFO	TiDB 127.0.0.1	[server.go:388] ["new connection"] [conn=294] [remoteAddr=127.0.0.1:56625]
2020-05-21 15:21:54	INFO	TiDB 127.0.0.1	[server.go:388] ["new connection"] [conn=292] [remoteAddr=127.0.0.1:56627]
2020-05-21 15:21:54	INFO	TiDB 127.0.0.1	[server.go:388] ["new connection"] [conn=293] [remoteAddr=127.0.0.1:56628]
2020-05-21 15:21:54	INFO	TiDB 127.0.0.1	[server.go:391] ["connection closed"] [conn=292]
2020-05-21 15:21:54	INFO	TiDB 127.0.0.1	[server.go:391] ["connection closed"] [conn=291]

Figure 423: Search result

This page consists of the following three areas:

- Parameter options (area 1 in the image above): These options are the same as the parameter options on the search homepage. You can re-select the parameters in the boxes and start a new search.
- Progress (area 2 in the image above): The current search progress is shown on the right side of this page, including the log search status and statistics of each node.
- Search results (area 3 in the image above):
 - Time: The time at which the log is generated. The time zone is the same as the time zone of the front-end user.
 - Level: log level.
 - Component: Shows the component name and its address.
 - Log: The body part of each log record, excluding the log time and log level. Logs that are too long are automatically truncated. Click a row to expand the full content. The full log can show up to 512 characters.

Note:

At most 500 search results can be previewed on this page. You can get the complete search results by downloading them.

Search progress

In the search progress area, a search on a node is called a search task. A search task might have the following statuses:

- Running: After starting the search, all tasks enter the **Running** status.
- Success: After the task is completed, it automatically enters the **Success** status. At this time, the logs have been cached in the local disk where the Dashboard backend is located, and can be provided to the frontend to download.
- Failed: When you cancel the search task, or the task exits with an error, the task enters the **Failed** status. When the task fails, the local temporary files are automatically cleaned.

The search progress area has the following three control buttons:

- **Download Selected:** Click this button to download logs of the selected components (only the completed ones can be selected), and you will get a tar file. Unzip this tar file to get one or more zip files (each component corresponds to a zip file). Unzip the zip file(s) to get the log text file.
- **Cancel:** Click this button to cancel all running tasks. You can click this button only when there are running tasks.
- **Retry:** Click this button to retry all failed tasks. You can click this button only when there are failed tasks and no running tasks.

11.6.1.11.3 Search history list

Click the **View search history** link on the log search homepage to enter page of search history list:



Preview and download logs by clicking Search

[View search history](#) ←

Figure 424: Search history entry

← Search Logs History					
Time Range	Level	Component	Keywords	State	Action
2020-05-19 14:52:54 ~ 2020-05-19...	INFO	1 TiDB, 1 TiKV, 1 PD	tidb	• Finished	Detail
2020-05-19 14:52:54 ~ 2020-05-19...	INFO	1 TiDB, 1 TiKV, 1 PD	memory	• Finished	Detail
2020-05-19 16:35:43 ~ 2020-05-19...	INFO	1 TiDB, 1 TiKV, 1 PD	worker	• Finished	Detail
2020-05-19 16:37:54 ~ 2020-05-19...	INFO	1 TiDB, 1 TiKV, 1 PD		• Finished	Detail
2020-05-19 16:37:59 ~ 2020-05-19...	INFO	1 TiDB, 1 TiKV, 1 PD	tidb	• Finished	Detail

Figure 425: Search history list

The history list shows the time range, log level, components, keywords, and search status of each search log. Click the **Detail** link in the **Action** column to see the search result details:

You can delete the search history that you no longer need. Click **Delete All** in the upper right corner, or select the rows to be deleted and then click **Delete selected** to delete the history:

Time Range		Level	Component	Keywords	State	Action
<input checked="" type="checkbox"/>	2020-05-19 14:52:54 ~ 2020-05-19...	INFO	1 TiDB, 1 TiKV, 1 PD	tidb	● Finished	Detail
<input checked="" type="checkbox"/>	2020-05-19 14:52:54 ~ 2020-05-19...	INFO	1 TiDB, 1 TiKV, 1 PD	memory	● Finished	Detail
<input checked="" type="checkbox"/>	2020-05-19 16:35:43 ~ 2020-05-19...	INFO	1 TiDB, 1 TiKV, 1 PD	worker	● Finished	Detail
	2020-05-19 16:37:54 ~ 2020-05-19...	INFO	1 TiDB, 1 TiKV, 1 PD		● Finished	Detail
	2020-05-19 16:37:59 ~ 2020-05-19...	INFO	1 TiDB, 1 TiKV, 1 PD	tidb	● Finished	Detail

Figure 426: Delete search history

11.6.1.12 [Instance Profiling]

11.6.1.12.1 Manual Instance Profiling Page

On the manual instance profiling page, you can collect continuous performance data of TiDB, TiKV, PD, and TiFlash instances without restarting any of them. The data collected can be displayed on a flame graph or a directed acyclic graph. The graph visually shows what internal operations are performed on the instances during the performance summary period and the corresponding proportions. With this graph, you can quickly learn the CPU resource consumption of these instances.

Access the page

You can access the instance profiling page using either of the following methods:

- After logging into TiDB Dashboard, click **Advanced Debugging > Profile Instances > Manually Profile** on the left navigation bar.

⌚ Key Visualizer

🛡️ Cluster Diagnostics

🔍 Search Logs

🛠️ Advanced Debugging

^

⌚ Profile Instances

^

Manually Profile

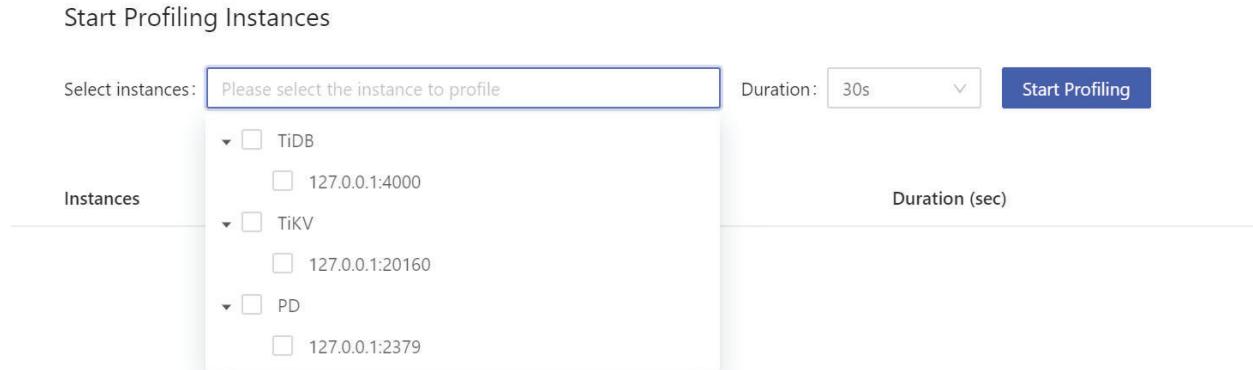
Continuous Profile

Figure 427: Access instance profiling page

- Visit http://127.0.0.1:2379/dashboard/#/instance_profiling in your browser. Replace 127.0.0.1:2379 with the actual PD instance address and port.

Start Profiling

In the instance profiling page, choose at least one target instance and click **Start Profiling** to start the instance profiling.



Start Profiling Instances

Select instances: Please select the instance to profile Duration: 30s Start Profiling

Instances	Duration (sec)
TiDB	
127.0.0.1:4000	
TiKV	
127.0.0.1:20160	
PD	
127.0.0.1:2379	

Figure 428: Start instance profiling

You can modify the profiling duration before starting the profiling. This duration is determined by the time needed for the profiling, which is 30 seconds by default. The 30-second duration takes approximately 30 seconds to complete.

View profiling status

After a profiling is started, you can view the profiling status and progress in real time.

[← History](#) Profiling Detail

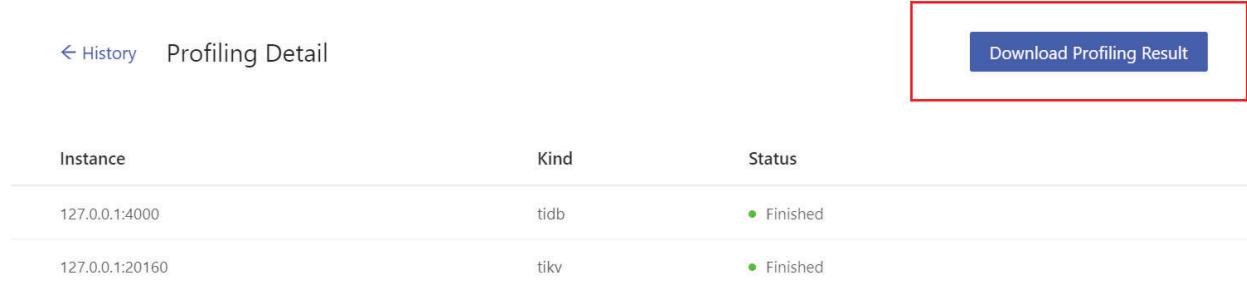
Instance	Kind	Status
127.0.0.1:4000	tidb	<div style="width: 27%;">27%</div>
127.0.0.1:20160	tikv	<div style="width: 27%;">27%</div>

Figure 429: Profiling detail

The profiling runs in the background. Refreshing or exiting the current page does not stop the profiling task that is running.

Download profiling result

After the profiling of all instances is completed, you can click **Download Profiling Result** in the upper right corner to download all profiling results.



The screenshot shows a table with three columns: Instance, Kind, and Status. There are two rows of data:

Instance	Kind	Status
127.0.0.1:4000	tidb	● Finished
127.0.0.1:20160	tikv	● Finished

A red box highlights the 'Download Profiling Result' button in the top right corner of the page header.

Figure 430: Download profiling result

You can also click a single instance on the list to directly view its profiling result.



The screenshot shows a table with three columns: Instance, Kind, and Status. There are two rows of data, with the first row highlighted by a red box:

Instance	Kind	Status
127.0.0.1:4000	tidb	● Finished
127.0.0.1:20160	tikv	● Finished

Figure 431: Single instance result

View profiling history

The profiling history is listed on the instance profiling page. Click one row of the list and you can view the status detail.

Start Profiling Instances

Select instances: Duration: Start Profiling

Instances	Status	Start At	Duration (sec)
1 TiDB, 1 TiKV	● Finished	Today at 5:09 PM	30
1 TiDB	● Finished	Today at 5:08 PM	30

Figure 432: View profiling history

For detailed operations on the profiling status page, see [View profiling status](#).

11.6.1.12.2 TiDB Dashboard Instance Profiling - Continuous Profiling

Warning:

Continuous Profiling is currently an experimental feature and is not recommended for use in production environments.

Introduced in TiDB 5.3.0, Continuous Profiling is a way to observe resource overhead at the system call level. With the support of Continuous Profiling, TiDB provides performance insight as clear as directly looking into the database source code, and helps R&D and operation and maintenance personnel to locate the root cause of performance problems using a flame graph.

With less than 0.5% performance loss, this feature takes continuous snapshots (similar to CT scan) of the database internal operations, turning the database from a “black box” into a “white box” that is more observable. This feature runs automatically after being enabled by one click and keeps storage results generated within the retention period. Storage results beyond the retention period are recycled to release the storage space.

Restrictions

Before enabling the Continuous Profiling feature, pay attention to the following restrictions:

- Under the x86 architecture, this feature supports TiDB, TiKV, and PD, but does not support TiFlash. This feature is not fully compatible with the ARM architecture and cannot be enabled under this architecture.

- Currently, this feature is only available for clusters deployed or upgraded using TiUP, and is unavailable for clusters deployed or upgraded by using TiDB Operator or binary packages.

Profiling content

With Continuous Profiling, you can collect continuous performance data of TiDB, TiKV, and PD instances, and have the nodes monitored day and night without restarting any of them. The data collected can be displayed on a flame graph or a directed acyclic graph. The graph visually shows what internal operations are performed on the instances during the performance summary period and the corresponding proportions. With this graph, you can quickly learn the CPU resource consumption of these instances.

Currently, Continuous Profiling can display the following performance data:

- TiDB/PD: CPU profile, Heap, Mutex, Goroutine (debug=2)
- TiKV: CPU profile

Enable Continuous Profiling

To enable Continuous Profiling, you need to first check the version information and then configure related settings on the control machine and TiDB Dashboard.

Check versions

Before enabling this feature, check the version of the TiUP cluster and ensure that it is 1.7.0 or above. If it is earlier than 1.7.0, upgrade it first.

1. Check the TiUP version:

```
tiup cluster --version
```

The command output shows the TiUP version, as shown below:

```
tiup version 1.7.0 tiup
Go Version: go1.17.2
Git Ref: v1.7.0
```

- If `tiup version` is earlier than 1.7.0, upgrade the TiUP cluster by referring to the next step.
- If `tiup version` is 1.7.0 or above, you can directly reload Prometheus.

2. Upgrade TiUP and TiUP cluster to the latest version.

- Upgrade TiUP:

```
tiup update --self
```

- Upgrade the TiUP cluster:

```
tiup update cluster
```

After TiUP and the TiUP cluster are upgraded to the latest version, version check is completed.

Configure the control machine and TiDB Dashboard

1. On the control machine, add the `ng_port` configuration item by using TiUP.

1. Open the configuration file of the cluster in the editing mode:

```
tiup cluster edit-config ${cluster-name}
```

2. Set parameters: add `ng_port:${port}` under `monitoring_servers`.

```
monitoring_servers:  
- host: 172.16.6.6  
  ng_port: ${port}
```

3. Reload Prometheus:

```
tiup cluster reload ${cluster-name} --role prometheus
```

After reloading Prometheus, you have finished operations on the control machine.

2. Enable the Continuous Profiling feature.

1. On TiDB Dashboard, click **Advanced Debugging > Profile Instances > Continuous Profile**.
 2. In the displayed window, switch on the button under **Enable Feature** on the right. Modify the value of **Retention Duration** as required or retain the default value.
 3. Click **Save** to enable this feature.

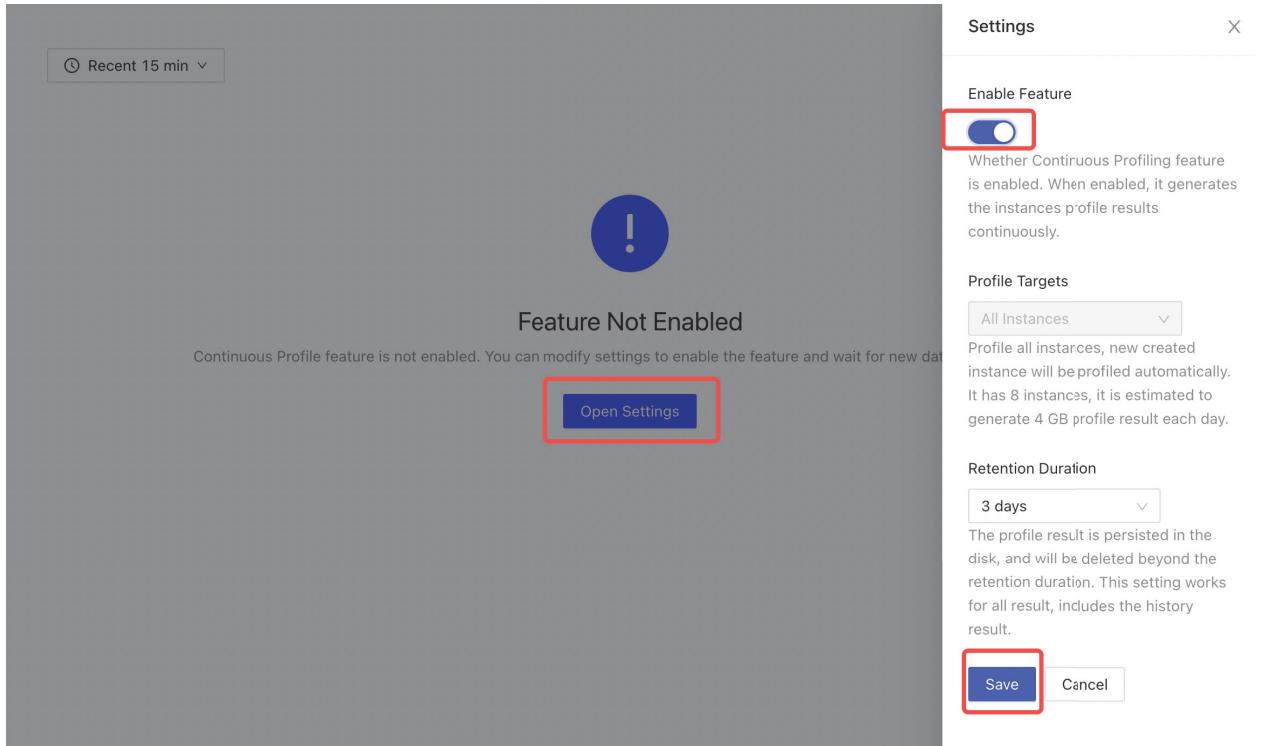


Figure 433: Enable the feature

Access the page

You can access the instance profiling page using either of the following methods:

- After logging into TiDB Dashboard, click **Advanced Debugging > Profile Instances > Continuous Profile** on the left navigation bar.

🛡 Cluster Diagnostics

🔍 Search Logs

⌚ Advanced Debugging ^

⌚ Profile Instances ^

Manually Profile

Continuous Profile

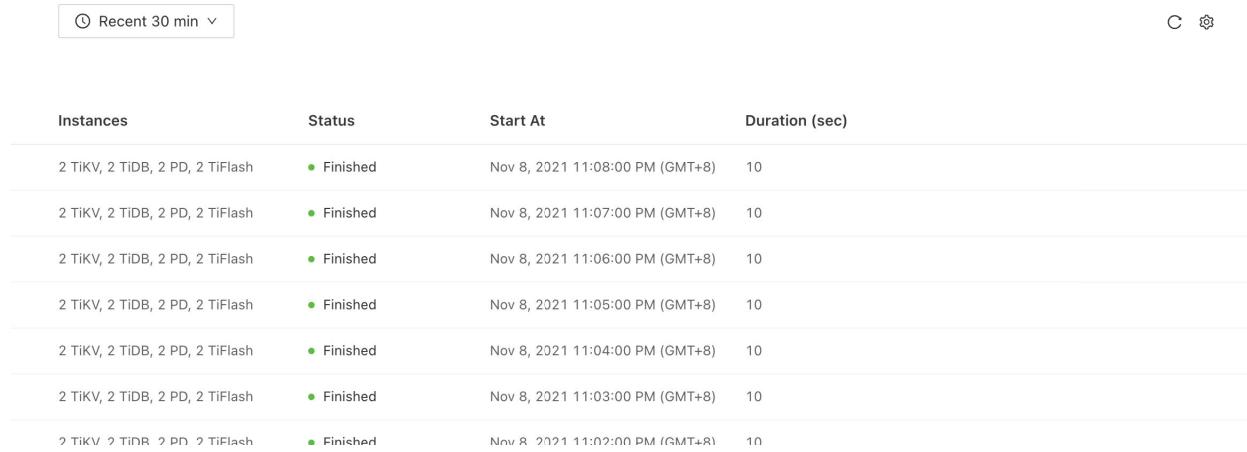
🔗 Debug Data

Figure 434: Access

- Visit http://127.0.0.1:2379/dashboard/#/continuous_profiling via your browser. Replace 127.0.0.1:2379 with the actual PD instance address and port.

View profiling history

After starting continuous profiling, you can view the profiling result on the instance profiling page.



Instances	Status	Start At	Duration (sec)
2 TiKV, 2 TiDB, 2 PD, 2 TiFlash	● Finished	Nov 8, 2021 11:08:00 PM (GMT+8)	10
2 TiKV, 2 TiDB, 2 PD, 2 TiFlash	● Finished	Nov 8, 2021 11:07:00 PM (GMT+8)	10
2 TiKV, 2 TiDB, 2 PD, 2 TiFlash	● Finished	Nov 8, 2021 11:06:00 PM (GMT+8)	10
2 TiKV, 2 TiDB, 2 PD, 2 TiFlash	● Finished	Nov 8, 2021 11:05:00 PM (GMT+8)	10
2 TiKV, 2 TiDB, 2 PD, 2 TiFlash	● Finished	Nov 8, 2021 11:04:00 PM (GMT+8)	10
2 TiKV, 2 TiDB, 2 PD, 2 TiFlash	● Finished	Nov 8, 2021 11:03:00 PM (GMT+8)	10
2 TiKV, 2 TiDB, 2 PD, 2 TiFlash	● Finished	Nov 8, 2021 11:02:00 PM (GMT+8)	10

Figure 435: Profiling history

Performance profiling runs in the background. Refreshing or exiting the current page will not terminate a running performance profiling task.

Download profiling result

On the profiling result page, you can click **Download Profiling Result** in the upper-right corner to download all profiling results.

[← History](#) Profiling Detail

Start At
Nov 8, 2021 11:08:00 PM (GMT+8)

[Download Profiling Result](#)

Instance	Component	Content	Status
172.16.5.36:10189	TiDB	CPU Profiling - 10s	● Finished
172.16.5.36:10189	TiDB	Heap	● Finished
172.16.5.36:10189	TiDB	Mutex	● Finished
172.16.5.36:10189	TiDB	Goroutine	● Finished
172.16.5.36:20292	TiFlash	CPU Profiling - 10s	● Finished
172.16.5.36:24180	TiKV	CPU Profiling - 10s	● Finished
172.16.5.36:24379	PD	CPU Profiling - 10s	● Finished

Figure 436: Download profiling result

You can also click an individual instance to view its profiling result:

Instance	Component	Content	Status
172.16.5.36:10189	TiDB	CPU Profiling - 10s	● Finished
172.16.5.36:10189	TiDB	Heap	● Finished
172.16.5.36:10189	TiDB	Mutex	● Finished
172.16.5.36:10189	TiDB	Goroutine	● Finished
172.16.5.36:20292	TiFlash	CPU Profiling - 10s	● Finished
172.16.5.36:24180	TiKV	CPU Profiling - 10s	● Finished
172.16.5.36:24379	PD	CPU Profiling - 10s	● Finished
172.16.5.36:24379	PD	Heap	● Finished
172.16.5.36:24379	PD	Goroutine	● Finished
172.16.5.36:24379	PD	Mutex	● Finished
172.16.5.40:10189	TiDB	Heap	● Finished

Figure 437: View the profiling result of an instance

Disable Continuous Profiling

1. On TiDB Dashboard, click **Advanced Debugging > Profile Instances > Continuous Profile** on the left navigation bar. Click **Settings**.

2. In the popped-up window, switch off the button under **Enable Feature**.
3. In the dialog box of **Disable Continuous Profiling Feature**, click **Disable**.
4. Click **Save**.

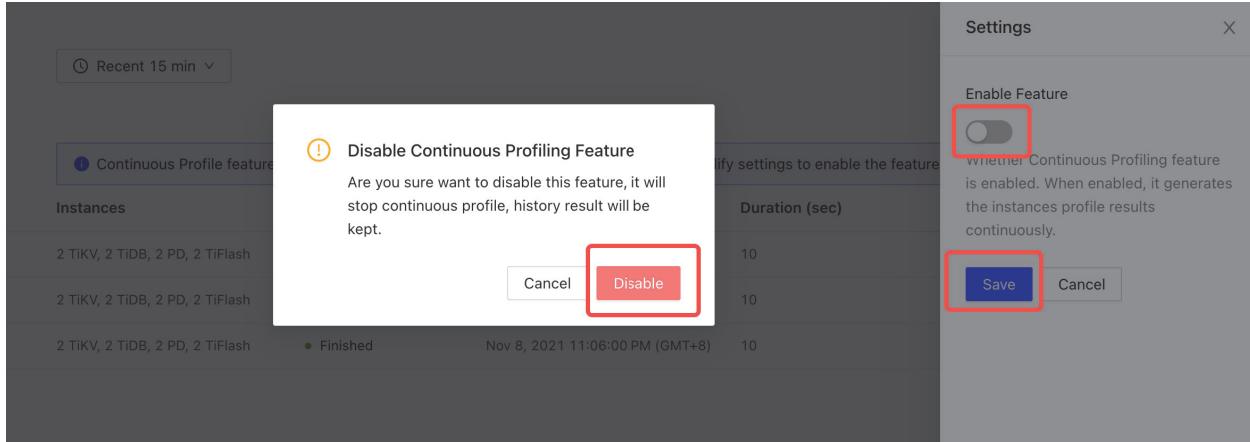


Figure 438: Disable the feature

11.6.1.13 Session Management and Configuration

11.6.1.13.1 Share TiDB Dashboard Sessions

You can share the current session of the TiDB Dashboard to other users so that they can access and operate the TiDB Dashboard without entering the user password.

Steps for the Inviter

1. Sign into TiDB Dashboard.
2. Click the username in the left sidebar to access the configuration page.
3. Click **Share Current Session**.

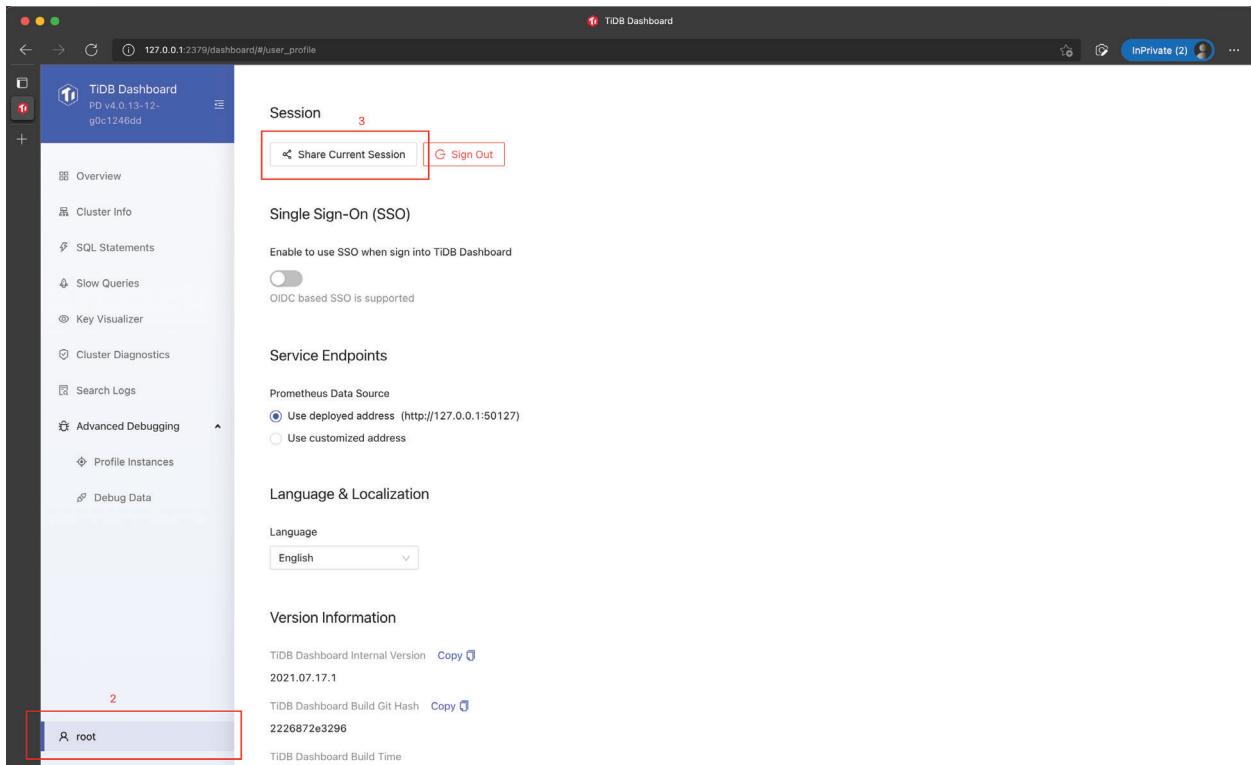


Figure 439: Sample Step

Note:

For security reasons, the shared session cannot be shared again.

4. Adjust sharing settings in the popup dialog:

- Expire in: How long the shared session will be effective. Signing out of the current session does not affect the effective time of the shared session.
- Share as read-only privilege: The shared session only permits read operations but not write operations (such as modifying configurations).

5. Click **Generate Authorization Code**.

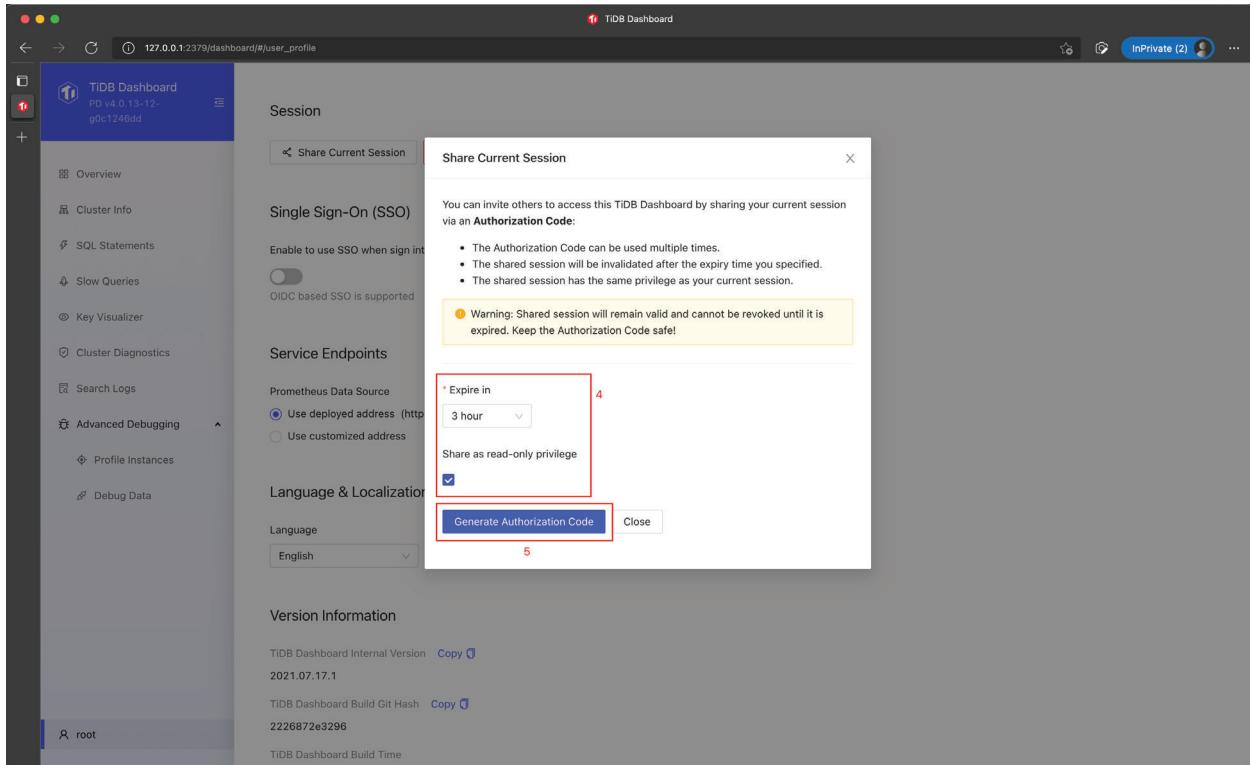


Figure 440: Sample Step

6. Provide the generated **Authorization Code** to the user to whom you want to share the session.

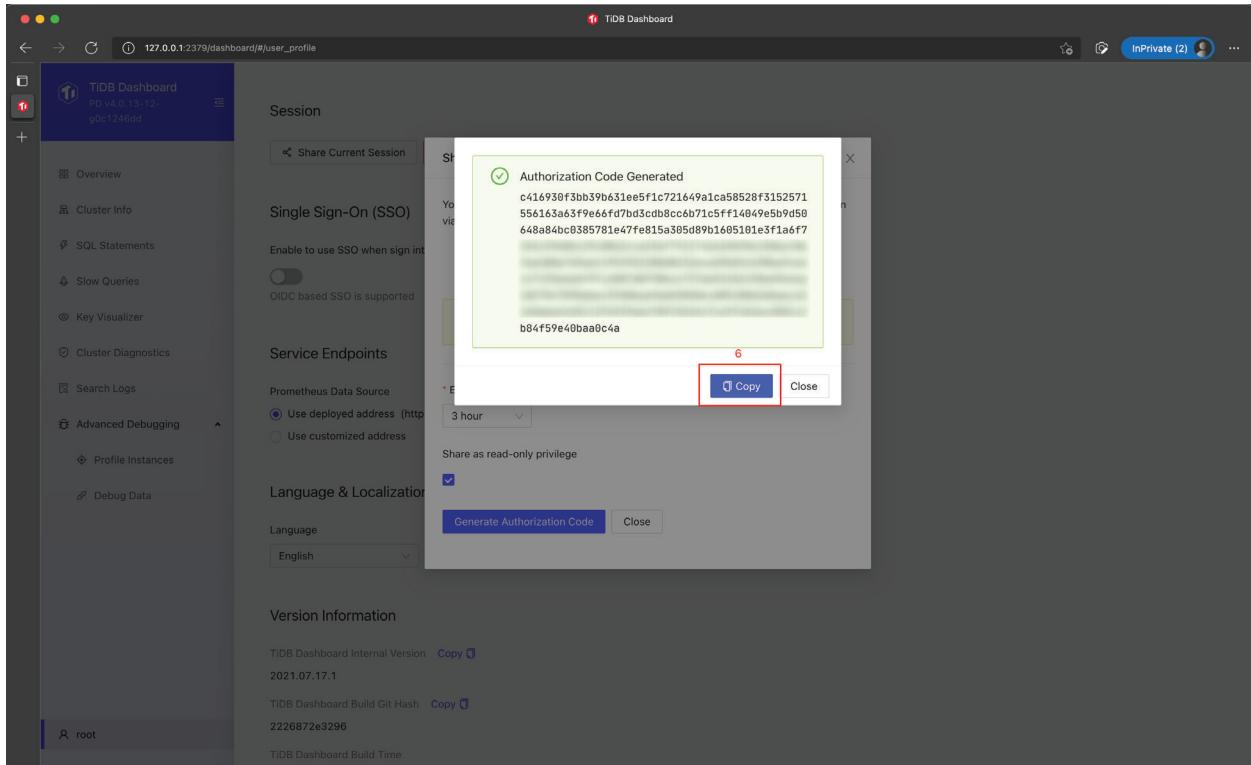


Figure 441: Sample Step

Warning:

Keep your authorization code secure and do not send it to anyone who is untrusted. Otherwise, they will be able to access and operate TiDB Dashboard without your authorization.

Steps for the Invitee

1. On the sign-in page of TiDB Dashboard, click **Use Alternative Authentication**.

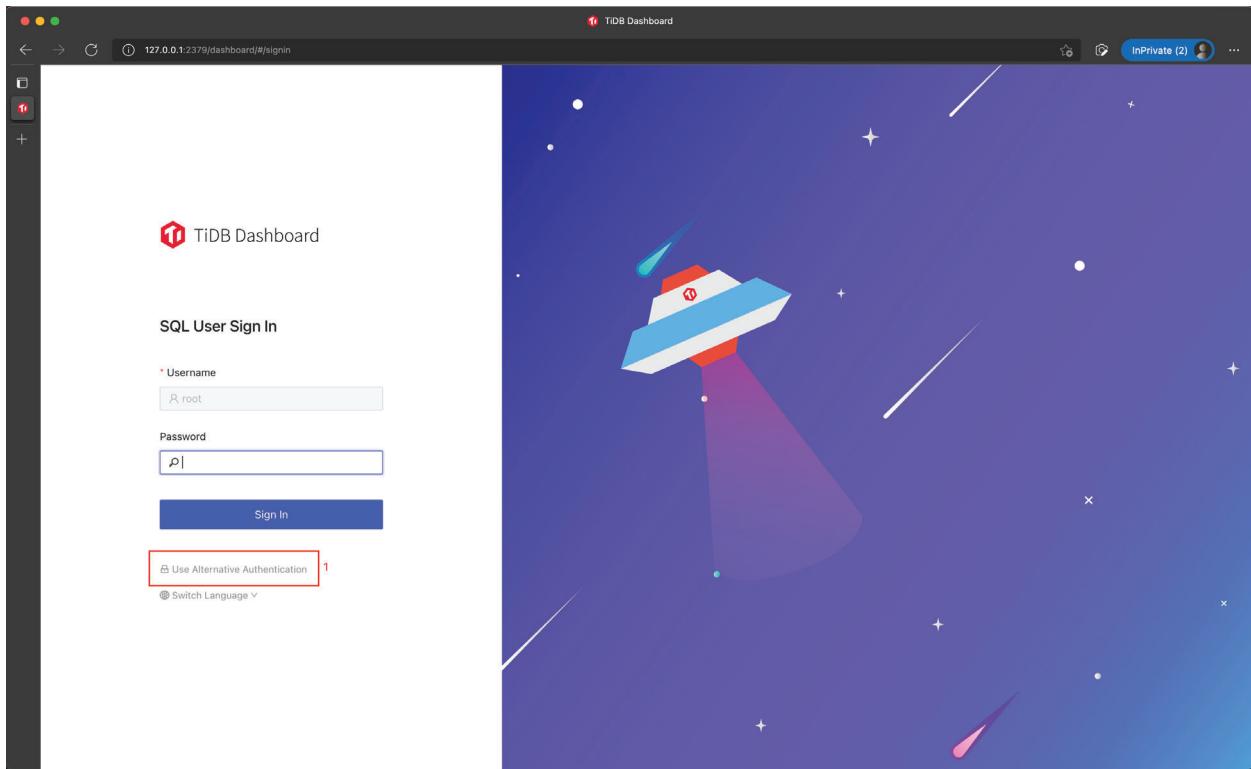


Figure 442: Sample Step

2. Click **Authorization Code** to use it to sign in.

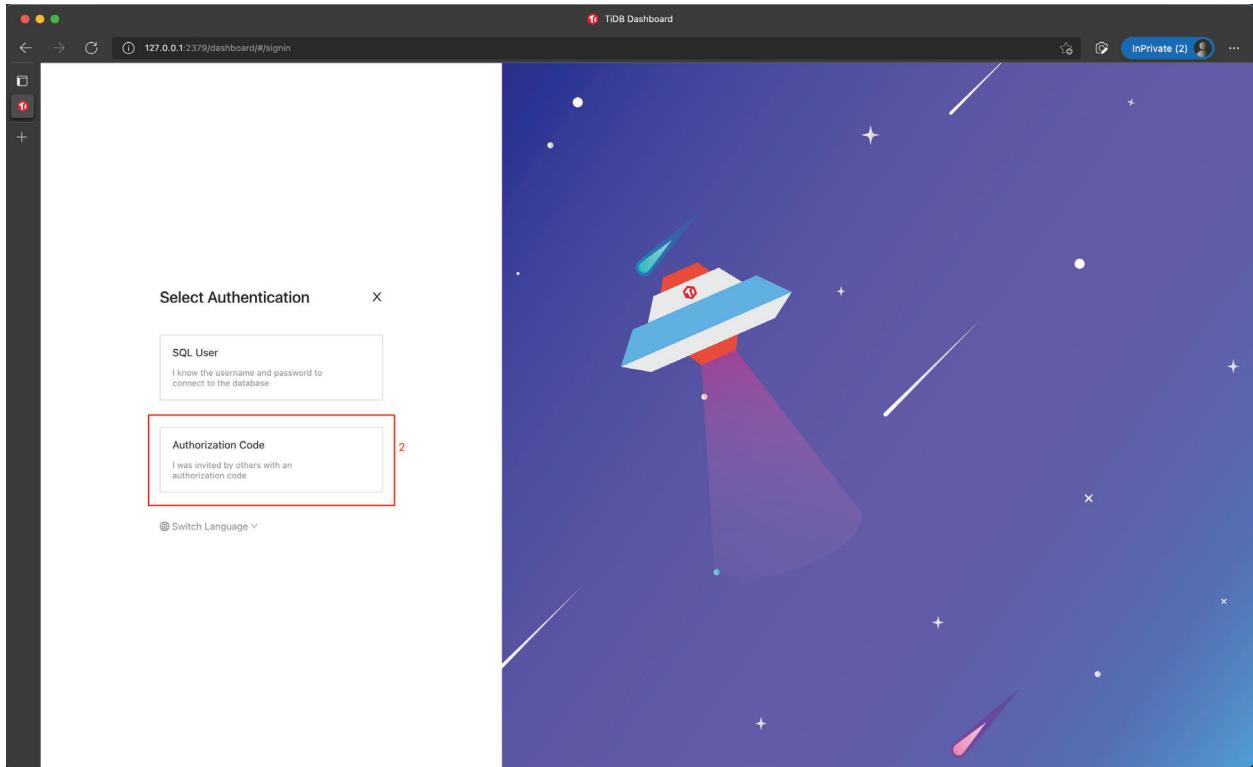


Figure 443: Sample Step

3. Enter the authorization code you have received from the inviter.
4. Click **Sign In**.

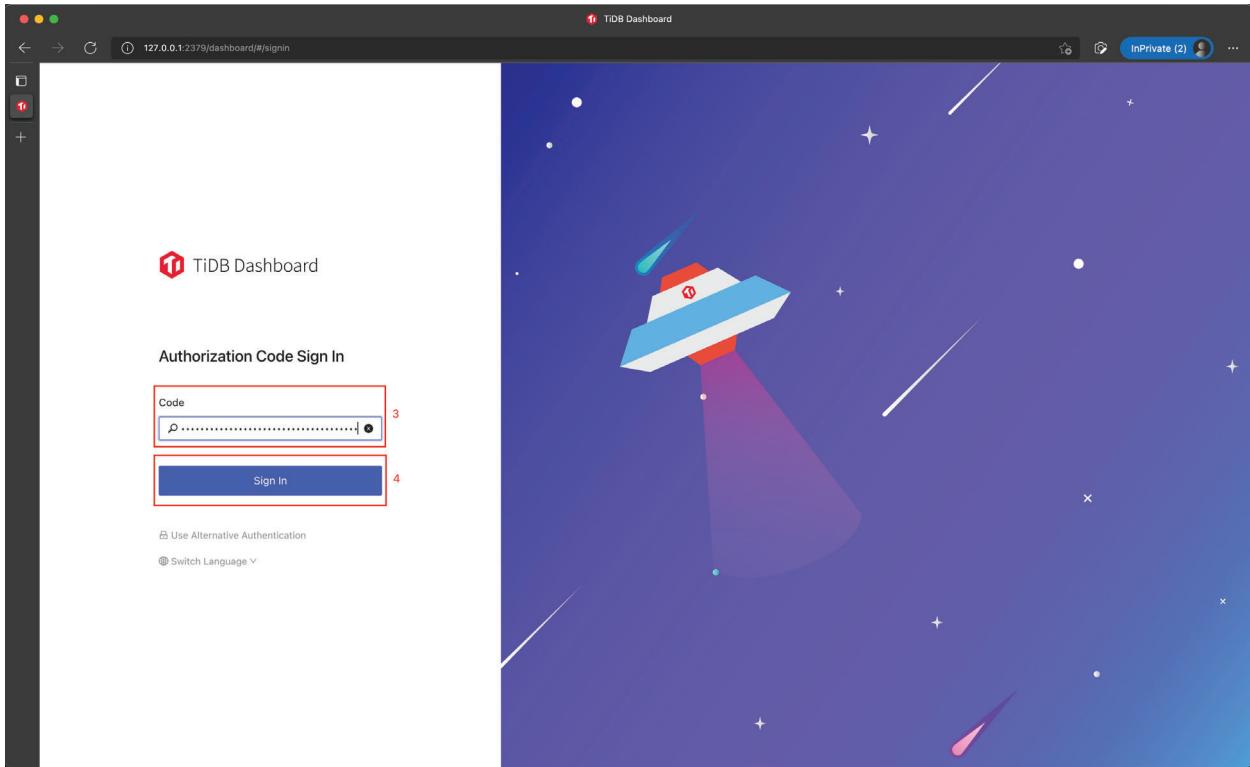


Figure 444: Sample Step

11.6.1.13.2 Configure SSO for TiDB Dashboard

TiDB Dashboard supports [OIDC](#)-based Single Sign-On (SSO). After enabling the SSO feature of TiDB Dashboard, the configured SSO service is used for your sign-in authentication and then you can access TiDB Dashboard without entering the SQL user password.

Configure OIDC SSO

Enable SSO

1. Sign into TiDB Dashboard.
2. Click the username in the left sidebar to access the configuration page.
3. In the **Single Sign-On** section, select **Enable to use SSO when sign into TiDB Dashboard**.
4. Fill the **OIDC Client ID** and the **OIDC Discovery URL** fields in the form.

Generally, you can obtain the two fields from the SSO service provider:

- OIDC Client ID is also called OIDC Token Issuer.
- OIDC Discovery URL is also called OIDC Token Audience.

5. Click **Authorize Impersonation** and input the SQL password.

TiDB Dashboard will store this SQL password and use it to impersonate a normal SQL sign-in after an SSO sign-in is finished.

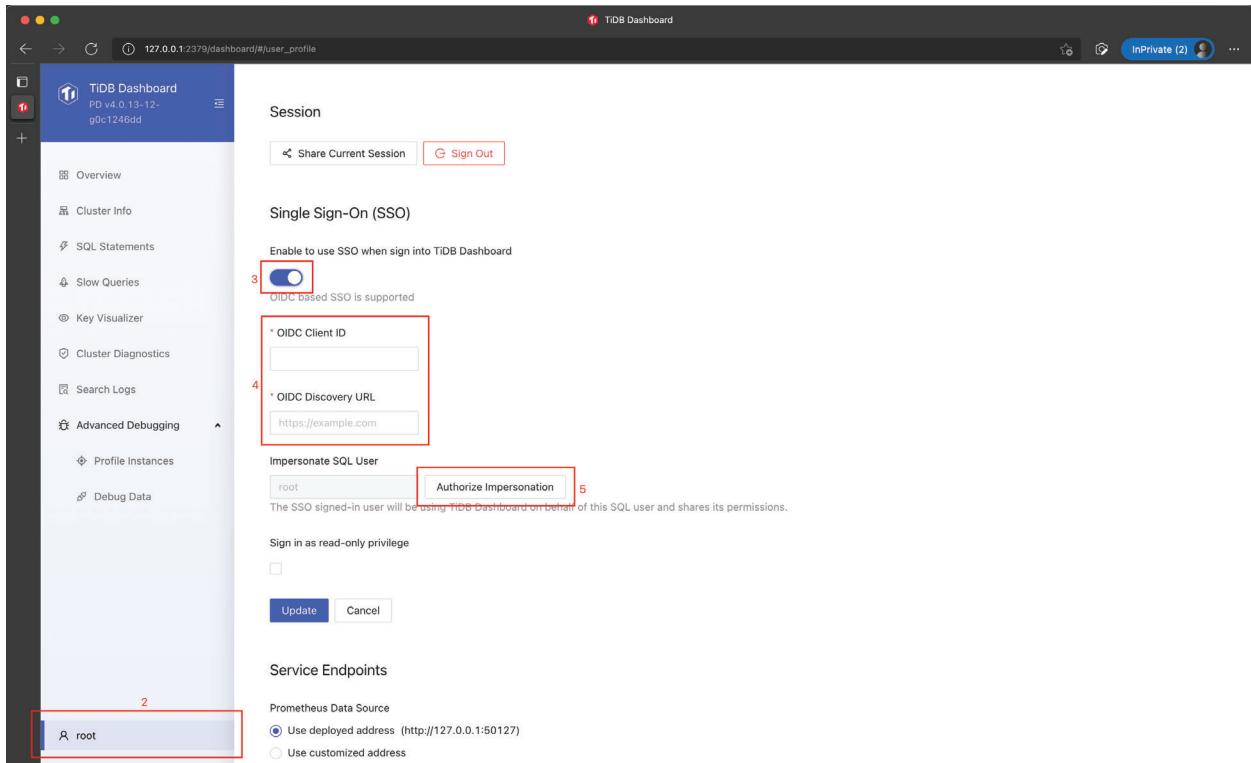


Figure 445: Sample Step

Note:

The password you have entered will be encrypted and stored. The SSO sign-in will fail after the password of the SQL user is changed. In this case, you can re-enter the password to bring SSO back.

6. Click **Authorize and Save**.

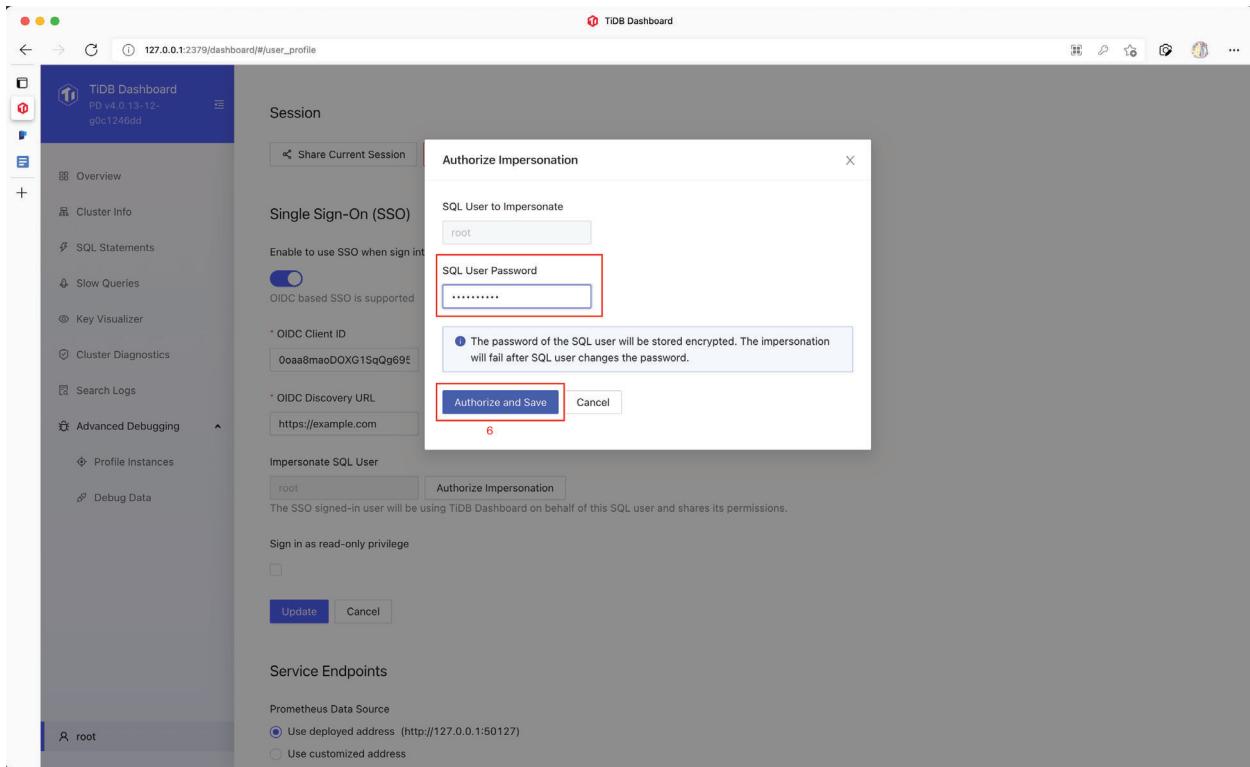


Figure 446: Sample Step

7. Click **Update** (Update) to save the configuration.

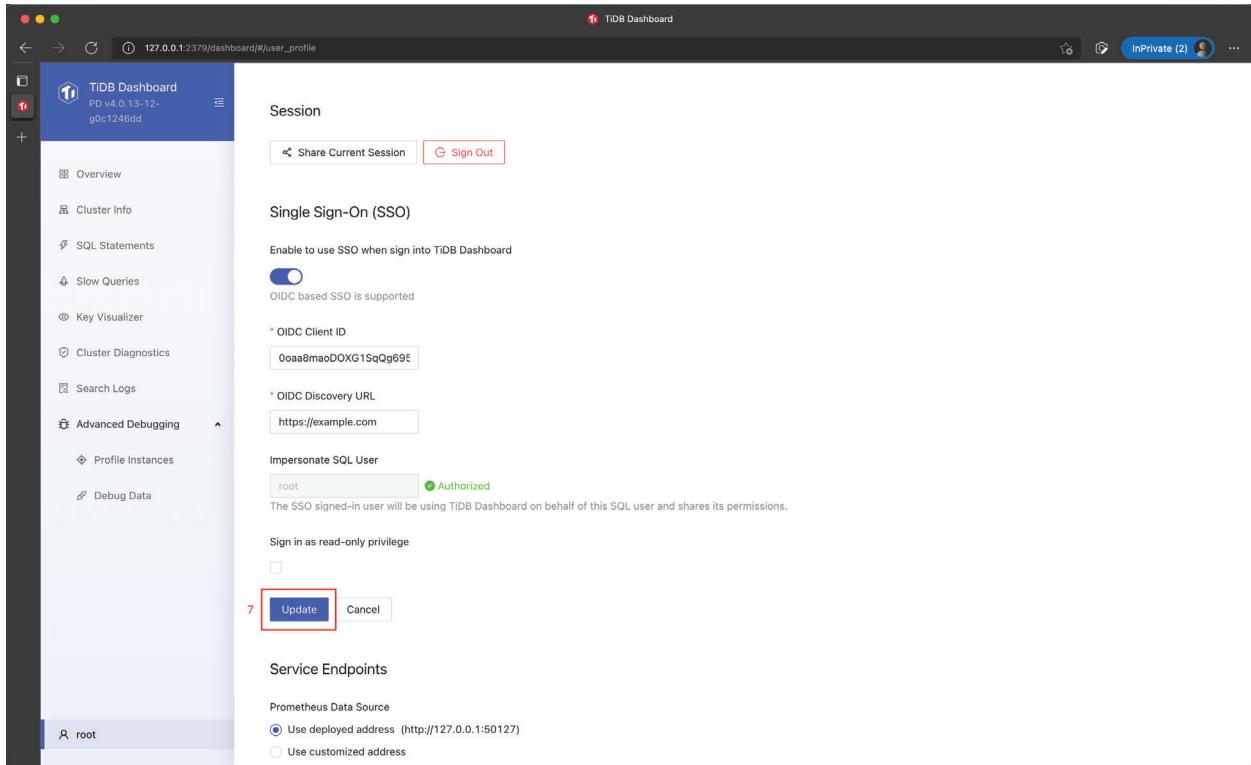


Figure 447: Sample Step

Now SSO sign-in has been enabled for TiDB Dashboard.

Note:

For security reasons, some SSO services require additional configuration for the SSO service, such as the trusted sign-in and sign-out URIs. Refer to the documentation of the SSO service for further information.

Disable SSO

You can disable the SSO, which will completely erase the stored SQL password:

1. Sign into TiDB Dashboard.
2. Click the username in the left sidebar to access the configuration page.
3. In the **Single Sign-On** section, deselect **Enable to use SSO when sign into TiDB Dashboard**.
4. Click **Update** (Update) to save the configuration.

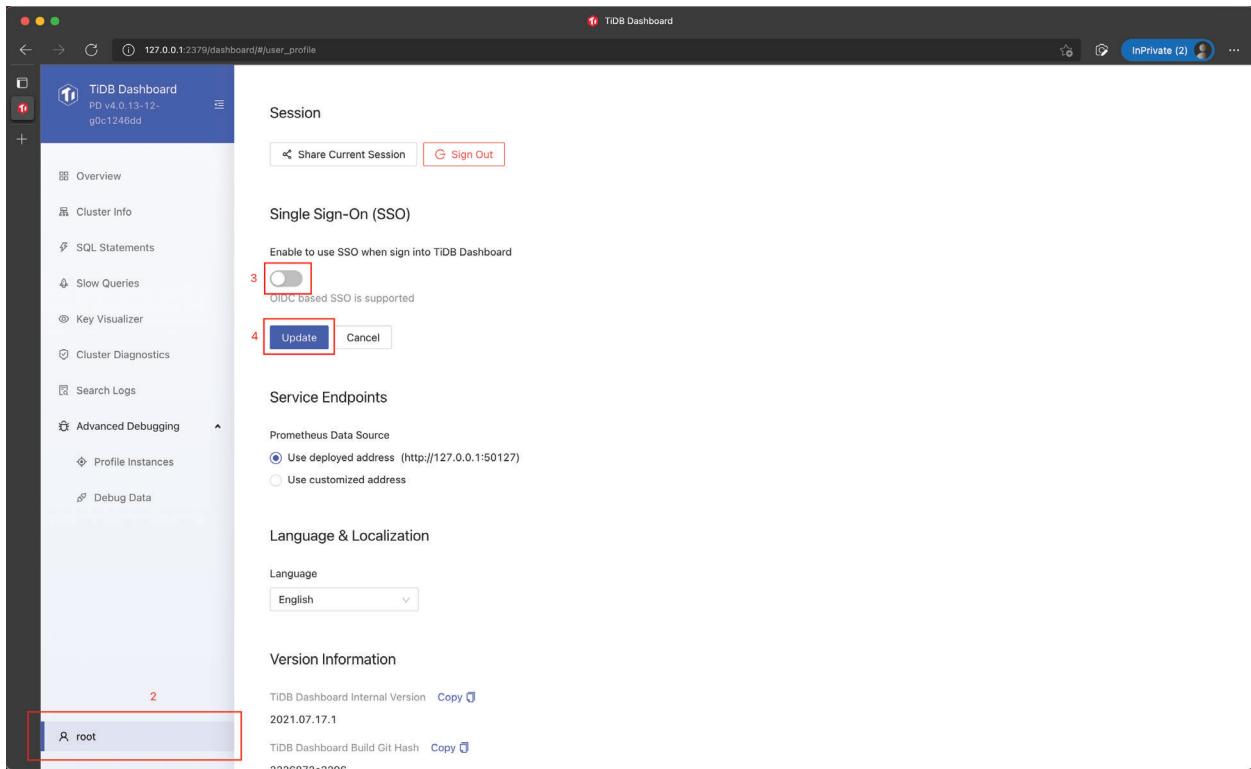


Figure 448: Sample Step

Re-enter the password after a password change

The SSO sign-in will fail once the password of the SQL user is changed. In this case, you can bring back the SSO sign-in by re-entering the SQL password:

1. Sign into TiDB Dashboard.
2. Click the username in the left sidebar to access the configuration page.
3. In the **Single Sign-On** section, Click **Authorize Impersonation** and input the updated SQL password.

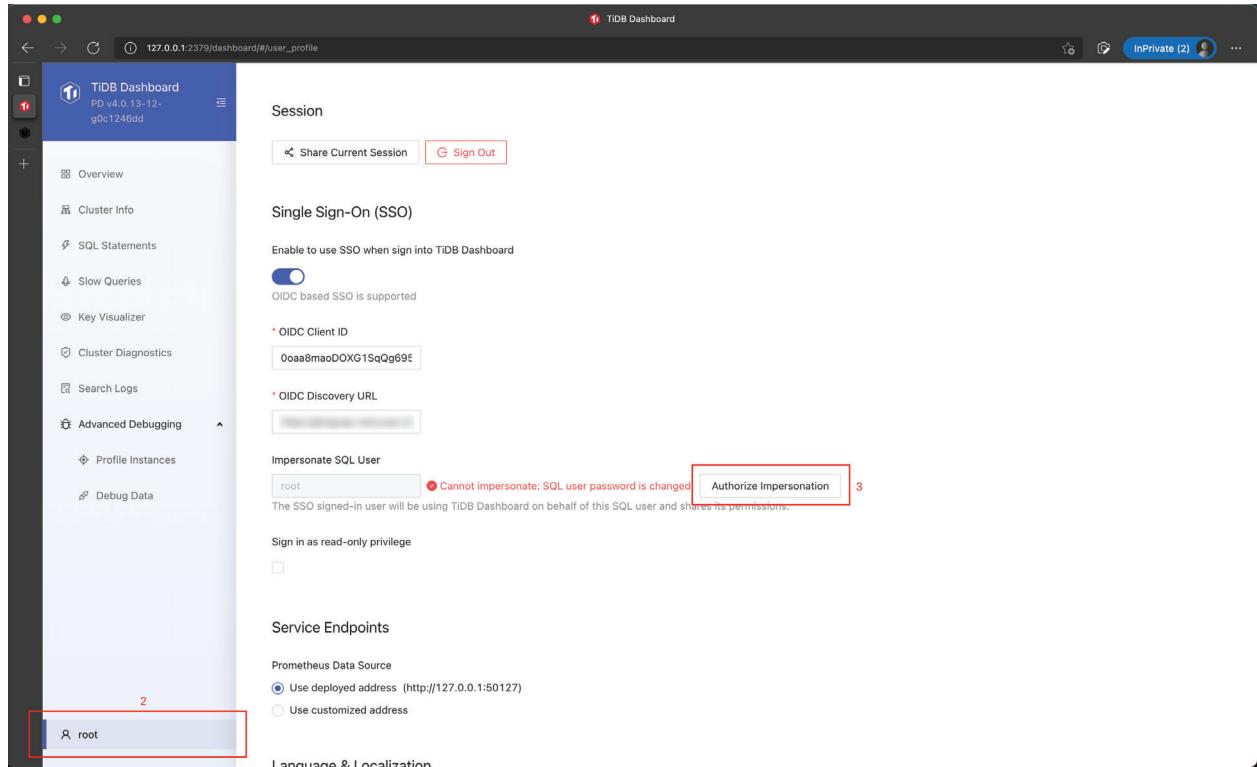


Figure 449: Sample Step

4. Click Authorize and Save.

Sign in via SSO

Once SSO is configured for TiDB Dashboard, you can sign in via SSO by taking following steps:

1. In the sign-in page of TiDB Dashboard, click **Sign in via Company Account**.

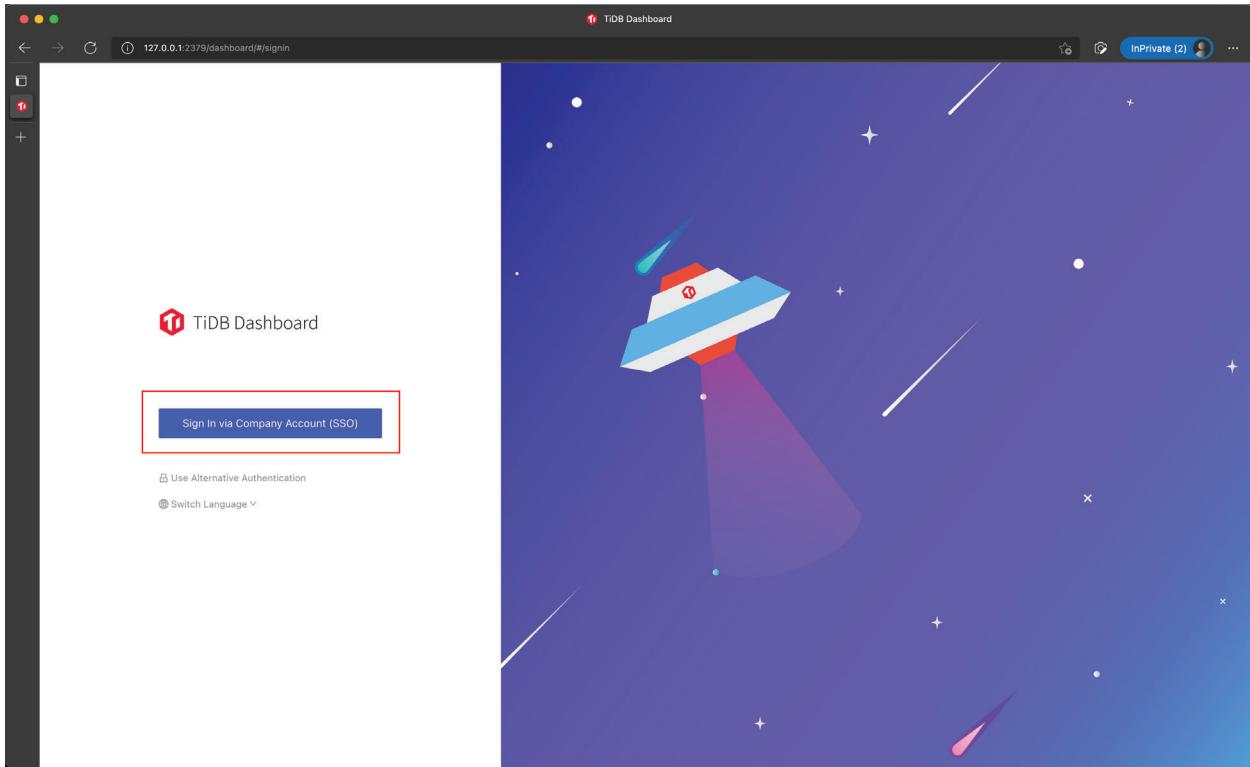


Figure 450: Sample Step

2. Sign into the system with SSO service configured.
3. You are redirected back to TiDB Dashboard to finish the sign-in.

Example 1: Use Okta for TiDB Dashboard SSO sign-in

[Okta](#) is an OIDC SSO identity service, which is compatible with the SSO feature of TiDB Dashboard. The steps below demonstrate how to configure Okta and TiDB Dashboard so that Okta can be used as the TiDB Dashboard SSO provider.

Step 1: Configure Okta

First, create an Okta Application Integration to integrate SSO.

1. Access the Okta administration site.
2. Navigate from the left sidebar **Applications > Applications**.
3. Click **Create App Integration**.

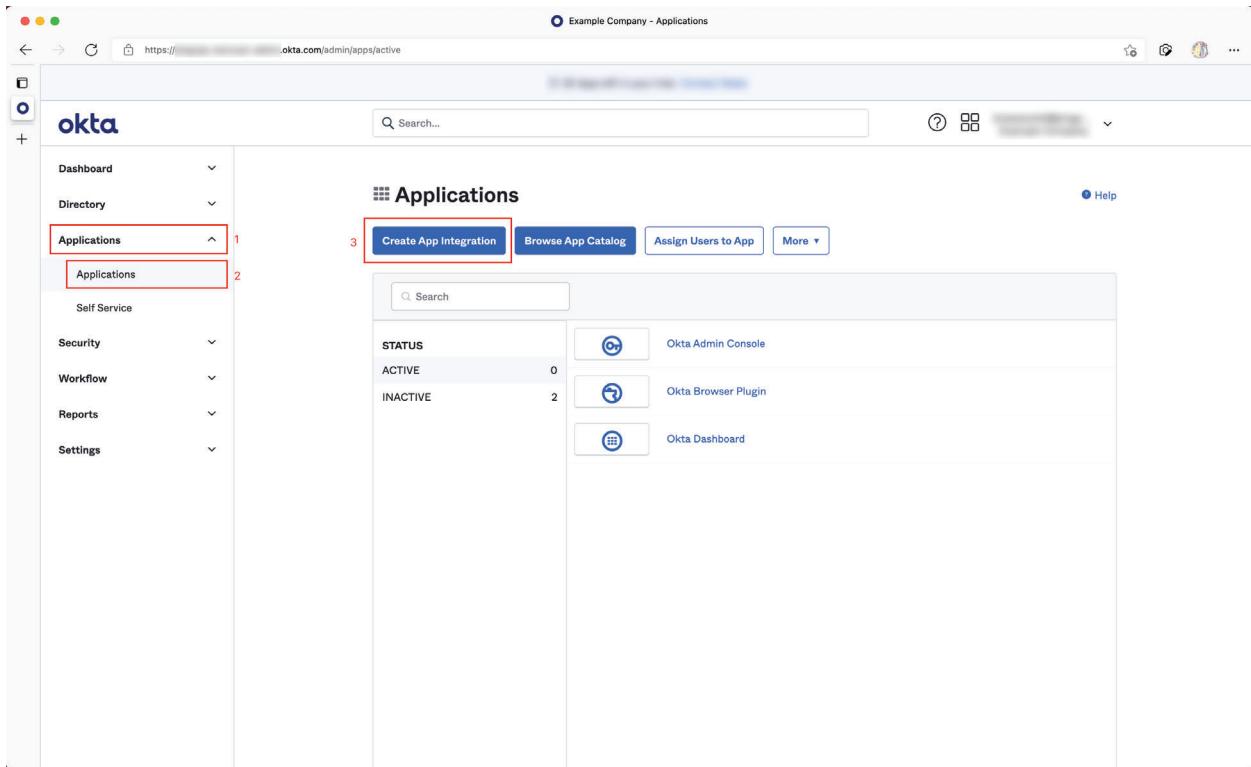


Figure 451: Sample Step

4. In the popped up dialog, choose **OIDC - OpenID Connect** in **Sign-in method**.
5. Choose **Single-Page Application** in **Application Type**.
6. Click the **Next** button.

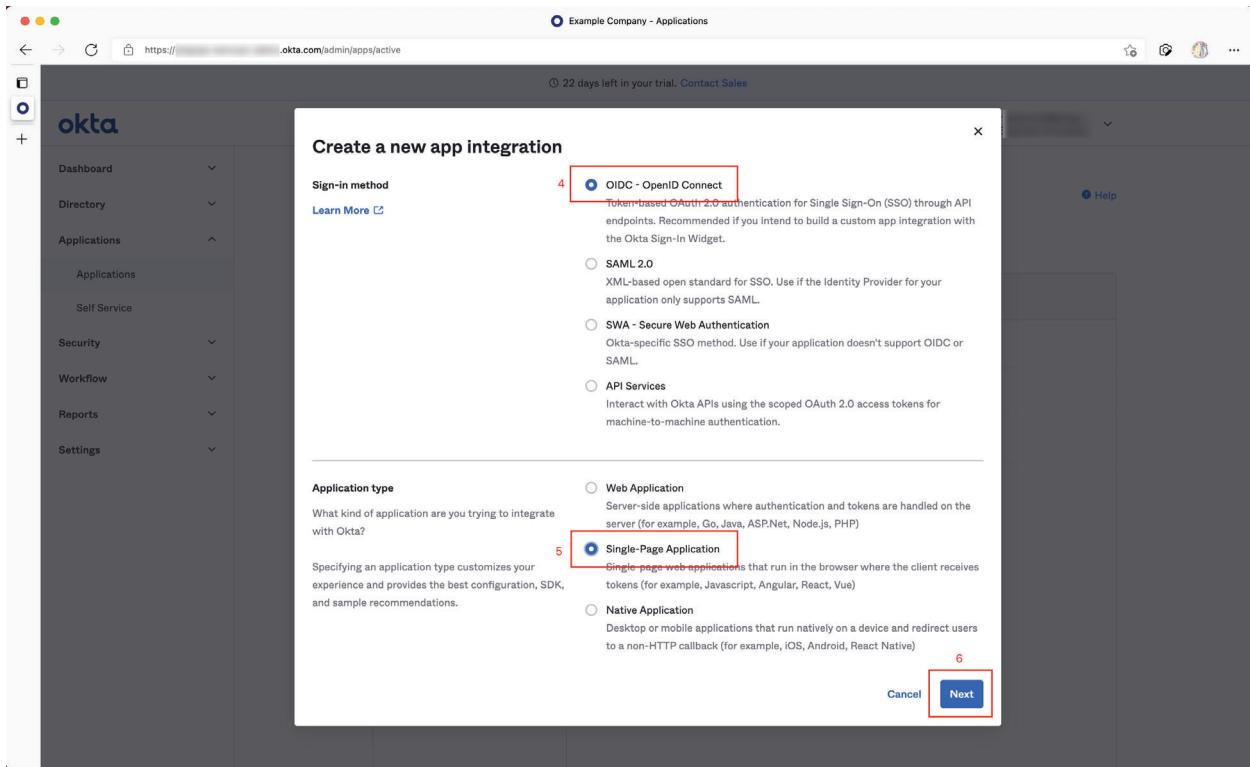


Figure 452: Sample Step

7. Fill **Sign-in redirect URIs** as follows:

```
http://DASHBOARD_IP:PORT/dashboard/?sso_callback=1
```

Substitute DASHBOARD_IP:PORT with the actual domain (or IP address) and port that you use to access the TiDB Dashboard in the browser.

8. Fill **Sign-out redirect URIs** as follows:

```
http://DASHBOARD_IP:PORT/dashboard/
```

Similarly, substitute DASHBOARD_IP:PORT with the actual domain (or IP address) and port.

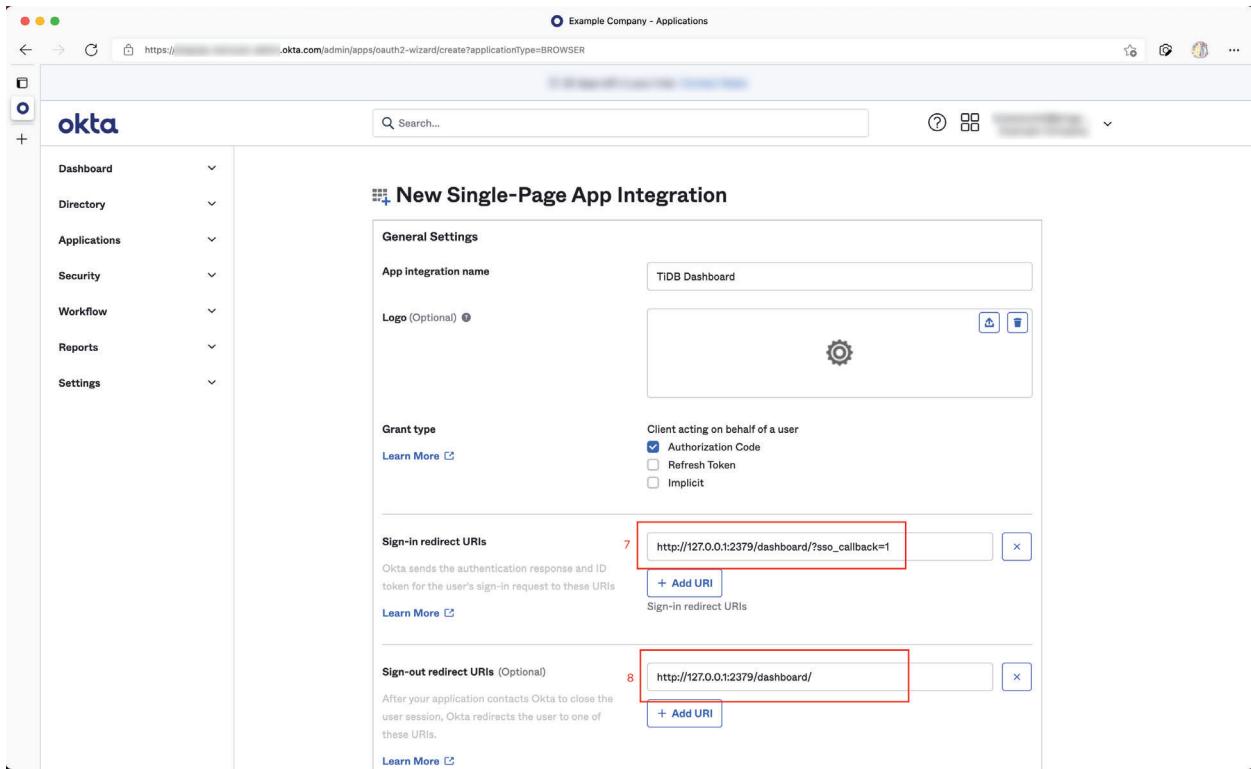


Figure 453: Sample Step

9. Configure what type of users in your organization is allowed for SSO sign-in in the **Assignments** field, and then click **Save** to save the configuration.

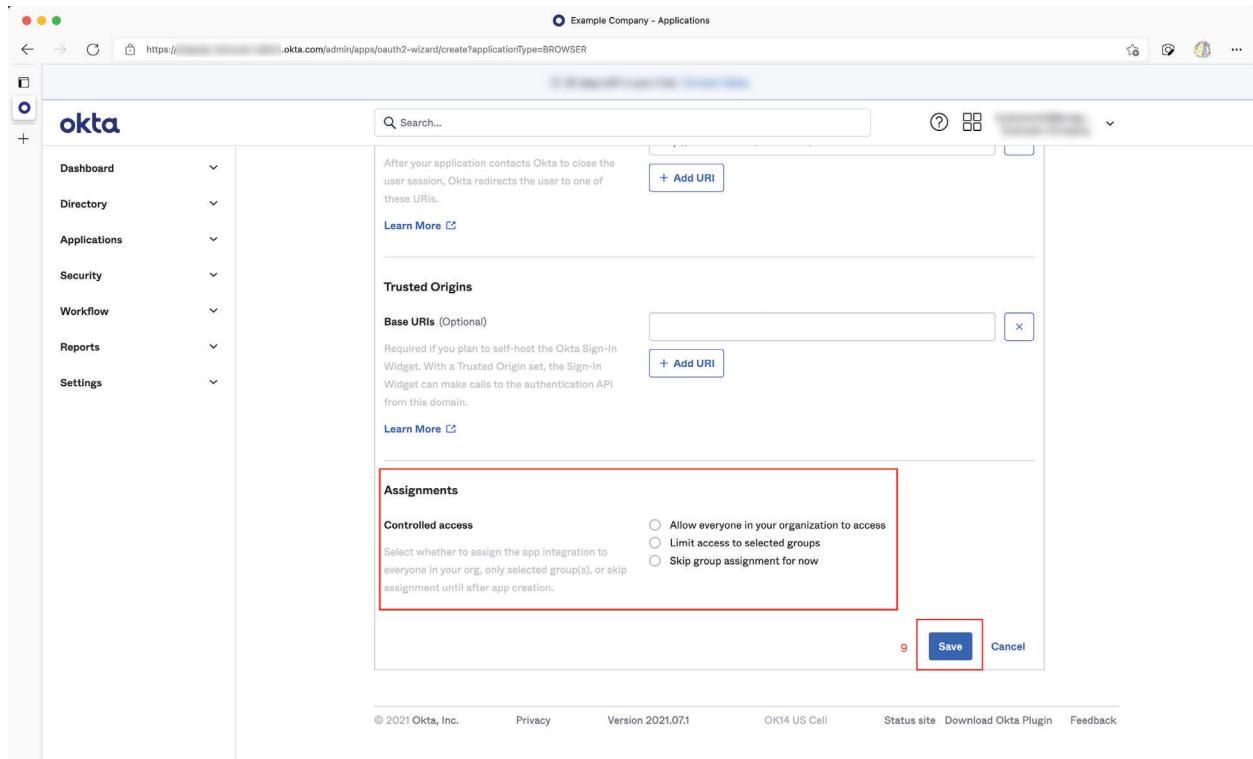
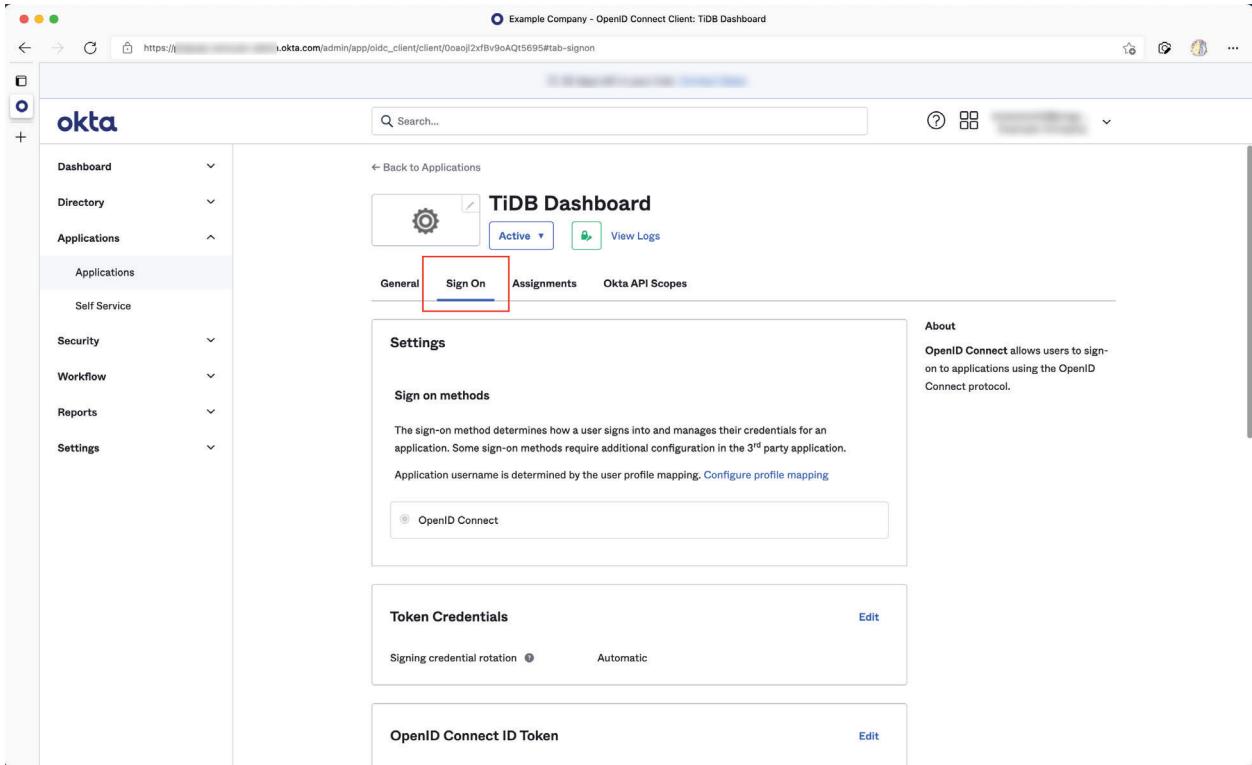


Figure 454: Sample Step

Step 2: Obtain OIDC information and fill in TiDB Dashboard

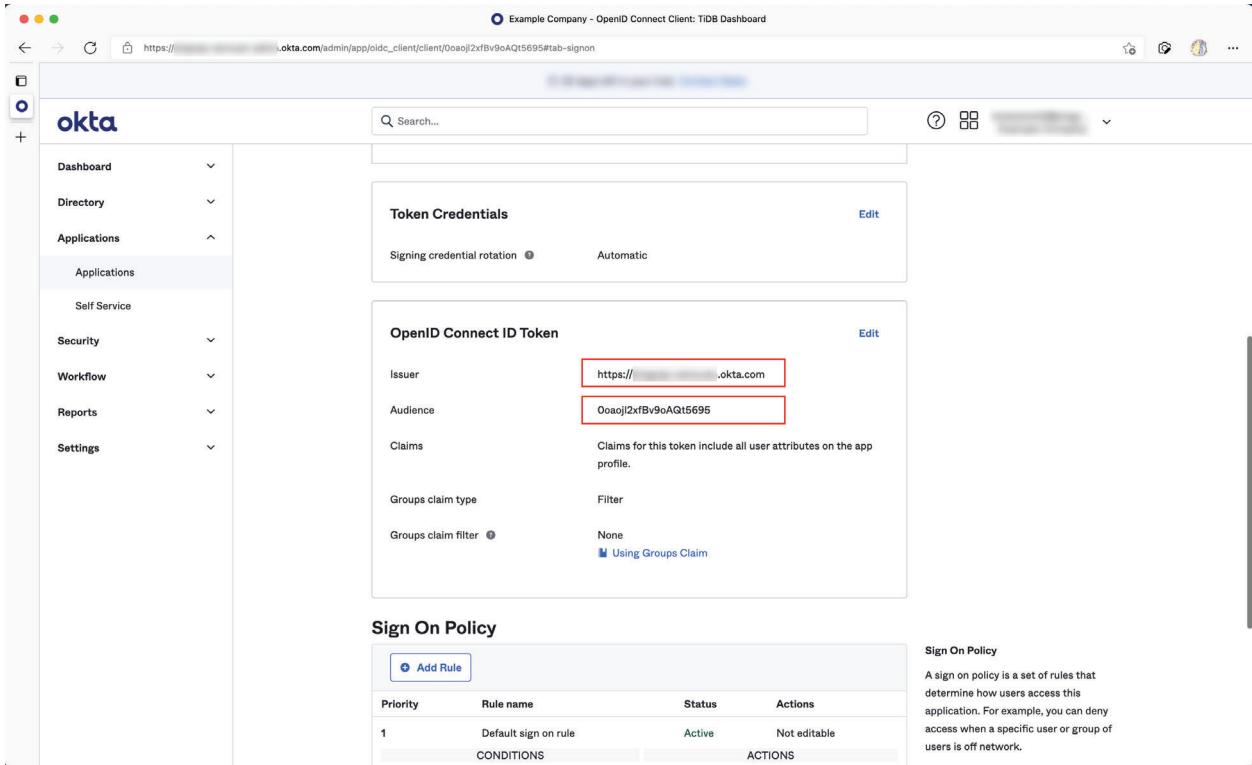
1. In the Application Integration just created in Okta, click **Sign On**.



The screenshot shows the Okta Admin Console interface for managing applications. The left sidebar has a tree view with 'Dashboard', 'Directory', 'Applications' (selected), 'Self Service', 'Security', 'Workflow', 'Reports', and 'Settings'. The main content area is titled 'TiDB Dashboard' and shows the application's configuration. The 'Sign On' tab is active, indicated by a red box. The 'Sign on methods' section contains a single option: 'OpenID Connect' (radio button selected). Below this are sections for 'Token Credentials' (with an 'Edit' link) and 'OpenID Connect ID Token' (with an 'Edit' link).

Figure 455: Sample Step 1

2. Copy values of the **Issuer** and **Audience** fields from the **OpenID Connect ID Token** section.



The screenshot shows the Okta Admin Console interface. On the left, there is a sidebar with navigation links: Dashboard, Directory, Applications (which is expanded), Applications, Self Service, Security, Workflow, Reports, and Settings. The main area has a search bar at the top. Below it, there are two main sections: "Token Credentials" and "OpenID Connect ID Token". Under "Token Credentials", the "Signing credential rotation" is set to "Automatic". Under "OpenID Connect ID Token", the "Issuer" is set to "https://[REDACTED].okta.com" and the "Audience" is set to "Ooaoj[REDACTED]5695". Both fields are highlighted with red boxes. Below these sections, there is a "Sign On Policy" section with a table and a detailed description.

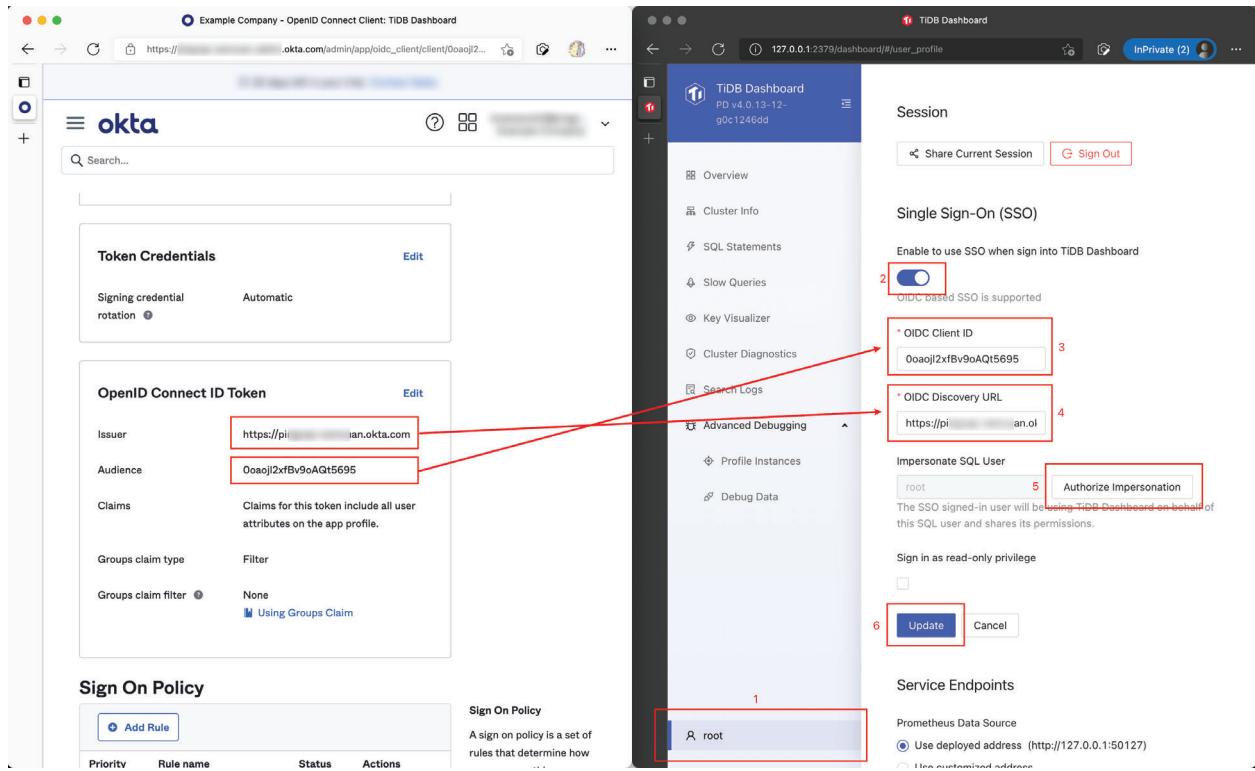
Add Rule			
Priority	Rule name	Status	Actions
1	Default sign on rule CONDITIONS	Active	Not editable ACTIONS

Sign On Policy

A sign on policy is a set of rules that determine how users access this application. For example, you can deny access when a specific user or group of users is off network.

Figure 456: Sample Step 2

3. Open the TiDB Dashboard configuration page, fill **OIDC Client ID** with **Issuer** obtained from the last step and fill **OIDC Discovery URL** with **Audience**. Then finish the authorization and save the configuration. For example:



The screenshot displays two browser windows side-by-side. The left window shows the Okta Admin Console with the 'Token Credentials' and 'OpenID Connect ID Token' sections highlighted. The right window shows the TiDB Dashboard configuration page under the 'Session' tab. Several fields are highlighted with red boxes and numbered 1 through 6:

- 1**: A dropdown menu in the TiDB Dashboard sidebar containing the option 'root'.
- 2**: A toggle switch labeled 'Enable to use SSO when sign into TiDB Dashboard'.
- 3**: The 'OIDC Client ID' field containing 'Ooaqlj2xfBv9oAQt5695'.
- 4**: The 'OIDC Discovery URL' field containing 'https://pi...ian.okta.com'.
- 5**: A checkbox labeled 'Sign in as read-only privilege'.
- 6**: A blue 'Update' button.

Figure 457: Sample Step 3

Now TiDB Dashboard has been configured to use Okta SSO for sign-in.

Example 2: Use Auth0 for TiDB Dashboard SSO sign-in

Similar to Okta, [Auth0](#) also provides OIDC SSO identity service. The following steps describe how to configure Auth0 and TiDB Dashboard so that Auth0 can be used as the TiDB Dashboard SSO provider.

Step 1: Configure Auth0

1. Access the Auth0 administration site.
2. Navigate on the left sidebar **Applications > Applications**.
3. Click **Create App Integration**.

Create application



Name *

TiDB Dashboard

You can change the application name later in the application settings.

Choose an application type

 Native Mobile, desktop, CLI and smart device apps running natively. e.g.: iOS, Electron, Apple TV apps	 Single Page Web Applications A JavaScript front-end app that uses an API. e.g.: Angular, React, Vue	 Regular Web Applications Traditional web app using redirects. e.g.: Node.js Express, ASP.NET, Java, etc.	 Machine to Machine Applications CLIs, daemons or services running on your backend. e.g.: Shell script
--	--	--	--

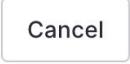


Figure 458: Create Application

In the popped-up dialog, fill **Name**, for example, "TiDB Dashboard".
→ Choose **Single Page Web Applications** in **Choose an application type**. Click **Create**.

4. Click **Settings**.

← Back to Applications

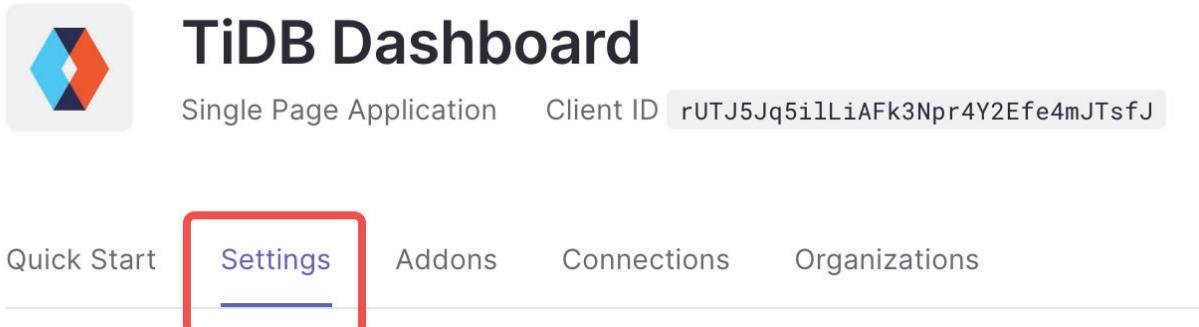


Figure 459: Settings

5. Fill **Allowed Callback URLs** as follows:

`http://DASHBOARD_IP:PORT/dashboard/?sso_callback=1`

Replace DASHBOARD_IP:PORT with the actual domain (or IP address) and port that you use to access the TiDB Dashboard in your browser.

6. Fill **Allowed Logout URLs** as follows:

`http://DASHBOARD_IP:PORT/dashboard/`

Similarly, replace DASHBOARD_IP:PORT with the actual domain (or IP address) and port.

Allowed Callback URLs

```
http://localhost:3001/dashboard/?sso_callback=1,  
http://127.0.0.1:2379/dashboard/?sso_callback
```

After the user authenticates we will only call back to any of these URLs. You can specify multiple valid URLs by comma-separating them (typically to handle different environments like QA or testing). Make sure to specify the protocol (`https://`) otherwise the callback may fail in some cases. With the exception of custom URI schemes for native clients, all callbacks should use protocol `https://`. You can use [Organization URL](#) parameters in these URLs.

Allowed Logout URLs

```
http://localhost:3001/dashboard/  
http://127.0.0.1:2379/dashboard/
```

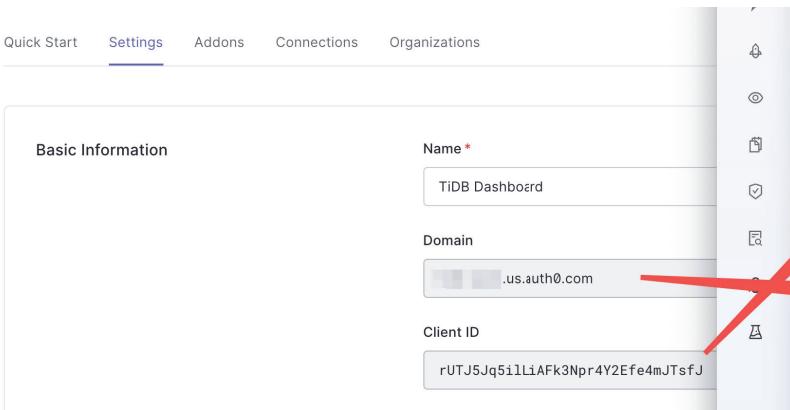
A set of URLs that are valid to redirect to after logout from Auth0. After a user logs out from Auth0 you can redirect them with the `returnTo` query parameter. The URL that you use in `returnTo` must be listed here. You can specify multiple valid URLs by comma-separating them. You can use the star symbol as a wildcard for subdomains (`*.google.com`). Query strings and hash information are not taken into account when validating these URLs. Read more about this at <https://auth0.com/docs/login/logout>

Figure 460: Settings

7. Keep the default values for other settings and click **Save Changes**.

Step 2: Obtain OIDC information and fill in TiDB Dashboard

1. Fill **OIDC Client ID** of TiDB Dashboard with **Client ID** in **Basic Information** under the **Settings** tab of Auth0.
2. Fill **OIDC Discovery URL** with the **Domain** field value prefixed with `https://` ↗ and suffixed with `/`, for example, `https://example.us.auth0.com/`. Complete authorization and save the configuration.



The screenshot shows the 'Settings' tab in the TiDB Dashboard. Under 'Basic Information', there are fields for 'Name' (TiDB Dashboard), 'Domain' (a placeholder domain), and 'Client ID' (a long string of characters). To the right, under 'Single Sign-On (SSO)', the 'Enable to use SSO when sign into TiDB Dashboard' toggle is turned on. Below it, 'OIDC based SSO is supported' is indicated. The 'OIDC Client ID' field contains the same value as the 'Client ID' in the basic information section. The 'OIDC Discovery URL' field contains a URL starting with 'https://'. A red arrow points from the 'Domain' field to the 'OIDC Client ID' field, highlighting the connection between them.

Figure 461: Settings

Now TiDB Dashboard has been configured to use Auth0 SSO for sign-in.

11.6.1.14 TiDB Dashboard FAQ

This document summarizes the frequently asked questions (FAQs) and answers about TiDB Dashboard.

11.6.1.14.1 Access-related FAQ

When the firewall or reverse proxy is configured, I am redirected to an internal address other than TiDB Dashboard

When multiple Placement Driver (PD) instances are deployed in a cluster, only one of the PD instances actually runs the TiDB Dashboard service. If you access other PD instances instead of this one, your browser redirects you to another address. If the firewall or reverse proxy is not properly configured for accessing TiDB Dashboard, when you visit the Dashboard, you might be redirected to an internal address that is protected by the firewall or reverse proxy.

- See [TiDB Dashboard Multi-PD Instance Deployment](#) to learn the working principle of TiDB Dashboard with multiple PD instances.
- See [Use TiDB Dashboard through a Reverse Proxy](#) to learn how to correctly configure a reverse proxy.
- See [Secure TiDB Dashboard](#) to learn how to correctly configure the firewall.

When TiDB Dashboard is deployed with dual network interface cards (NICs), TiDB Dashboard cannot be accessed using another NIC

For security reasons, TiDB Dashboard on PD only monitors the IP addresses specified during deployment (that is, it only listens on one NIC), not on 0.0.0.0. Therefore, when

multiple NICs are installed on the host, you cannot access TiDB Dashboard using another NIC.

If you have deployed TiDB using the `tiup cluster` or `tiup playground` command, currently this problem cannot be solved. It is recommended that you use a reverse proxy to safely expose TiDB Dashboard to another NIC. For details, see [Use TiDB Dashboard behind a Reverse Proxy](#).

11.6.1.14.2 UI-related FAQ

A `prometheus_not_found` error is shown in **QPS** and **Latency** sections on the Overview page

The **QPS** and **Latency** sections on the **Overview** page require a cluster with Prometheus deployed. Otherwise, the error is shown. You can solve this problem by deploying a Prometheus instance in the cluster.

If you still encounter this problem when the Prometheus instance has been deployed, the possible reason is that your deployment tool is out of date (TiUP or TiDB Operator), and your tool does not automatically report metrics addresses, which makes TiDB Dashboard unable to query metrics. You can upgrade your deployment tool to the latest version and try again.

If your deployment tool is TiUP, take the following steps to solve this problem. For other deployment tools, refer to the corresponding documents of those tools.

1. Upgrade TiUP and TiUP Cluster:

```
tiup update --self  
tiup update cluster --force
```

2. After the upgrade, when a new cluster is deployed with Prometheus instances, the metrics can be displayed normally.
3. After the upgrade, for an existing cluster, you can restart this cluster to report the metrics addresses. Replace `CLUSTER_NAME` with the actual cluster name:

```
tiup cluster start CLUSTER_NAME
```

Even if the cluster has been started, still execute this command. This command does not affect the normal application in the cluster, but refreshes and reports the metrics addresses, so that the monitoring metrics can be displayed normally in TiDB Dashboard.

An `invalid connection` error is shown in **Top SQL Statements** and **Recent Slow Queries** on the Overview page

The possible reason is that you have enabled the `prepared-plan-cache` feature of TiDB. As an experimental feature, when enabled, `prepared-plan-cache` might not function properly in specific TiDB versions, which could cause this problem in TiDB Dashboard (and other

applications). You can disable `prepared-plan-cache` by updating [TiDB Configuration file](#) to solve this problem.

An `unknown field` error is shown in **Slow Queries** page

If the `unknown field` error appears on the **Slow Queries** page after the cluster upgrade, the error is related to a compatibility issue caused by the difference between TiDB Dashboard server fields (which might be updated) and user preferences fields (which are in the browser cache). This issue has been fixed. If your cluster is earlier than v5.0.3 or v4.0.14, perform the following steps to resolve the issue:

To clear your browser cache, take the following steps:

1. Open TiDB Dashboard page.
2. Open Developer Tools. Different browsers have different ways of opening Developer Tools. After clicking the **Menu Bar**:
 - Firefox: Menu Web Developer Toggle Tools, or Tools Web Developer Toggle Tools.
 - Chrome: More tools Developer tools.
 - Safari: Develop Show Web Inspector. If you can't see the Develop menu, go to Safari Preferences Advanced, and check the Show Develop menu in menu bar checkbox.

In the following example, Chrome is used.

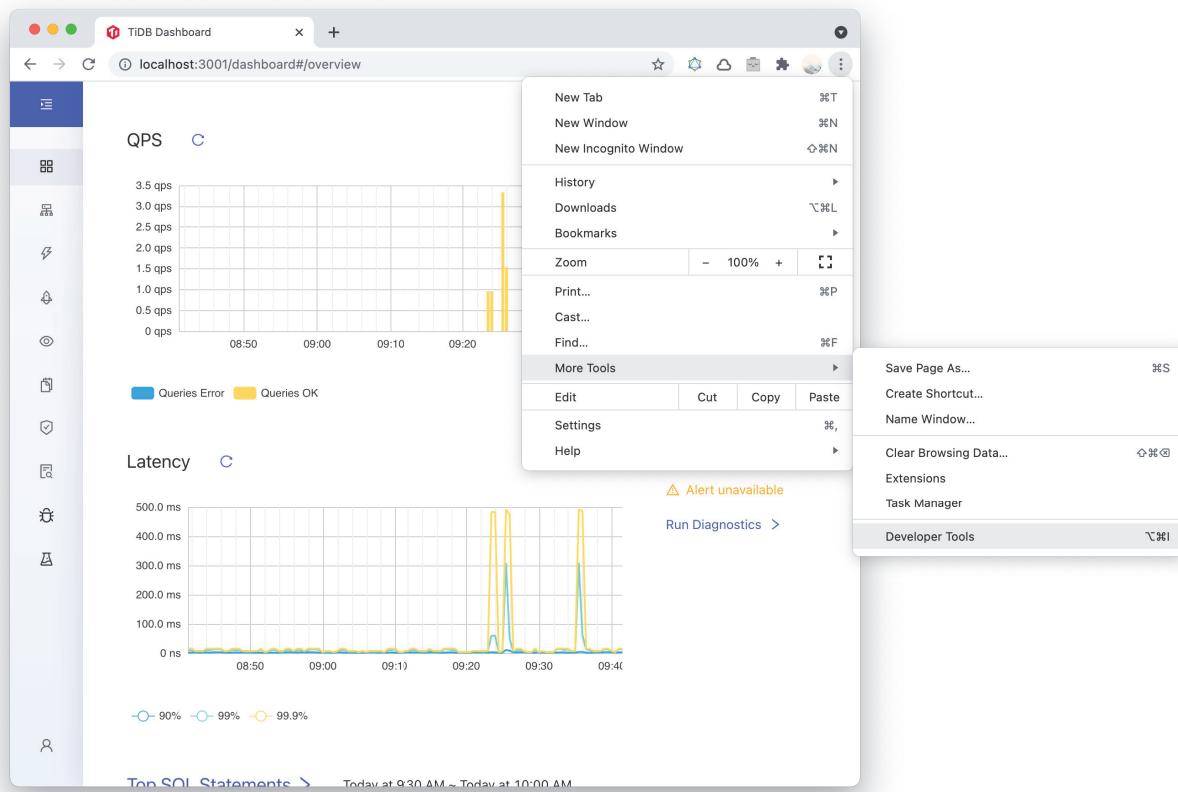


Figure 462: Opening DevTools from Chrome's main menu

3. Select the **Application** panel, expand the **Local Storage** menu and select the **TiDB Dashboard** page domain. Click the **Clear All** button.

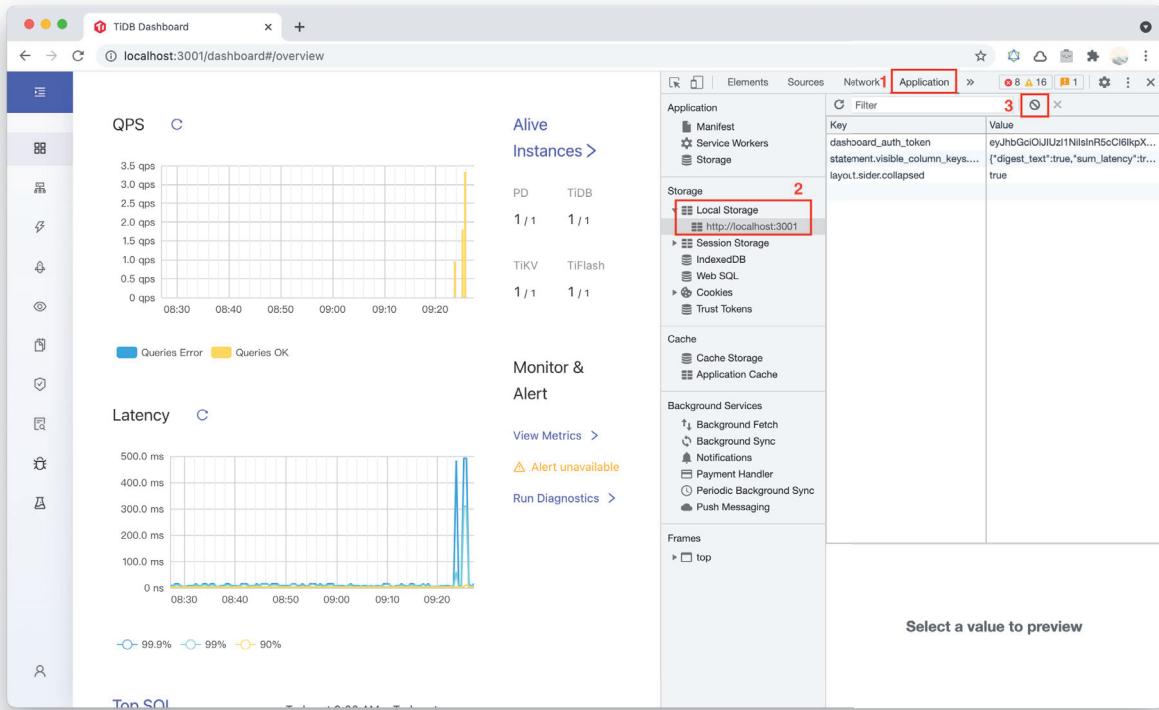


Figure 463: Clear the Local Storage

11.7 CLI

11.7.1 TiKV Control User Guide

TiKV Control (`tikv-ctl`) is a command line tool of TiKV, used to manage the cluster. Its installation directory is as follows:

- If the cluster is deployed using TiUP, `tikv-ctl` directory is in the in `~/.tiup/components/ctl/{VERSION}/` directory.

11.7.1.1 Use TiKV Control in TiUP

Note:

It is recommended that the version of the Control tool you use is consistent with the version of the cluster.

tikv-ctl is also integrated in the tiup command. Execute the following command to call the tikv-ctl tool:

```
tiup ctl tikv
```

```
Starting component `ctl`: /home/tidb/.tiup/components/ctl/v4.0.8/ctl tikv
TiKV Control (tikv-ctl)
Release Version: 4.0.8
Edition: Community
Git Commit Hash: 83091173e960e5a0f5f417e921a0801d2f6635ae
Git Commit Branch: heads/refs/tags/v4.0.8
UTC Build Time: 2020-10-30 08:40:33
Rust Version: rustc 1.42.0-nightly (0de96d37f 2019-12-19)
Enable Features: jemalloc mem-profiling portable sse protobuf-codec
Profile: dist_release

A tool for interacting with TiKV deployments.

USAGE:
    TiKV Control (tikv-ctl) [FLAGS] [OPTIONS] [SUBCOMMAND]

FLAGS:
    -h, --help                  Prints help information
    --skip-paranoid-checks     Skip paranoid checks when open rocksdb
    -V, --version                Prints version information

OPTIONS:
    --ca-path <ca_path>          Set the CA certificate path
    --cert-path <cert_path>        Set the certificate path
    --config <config>              Set the config for rocksdb
    --db <db>                      Set the rocksdb path
    --decode <decode>                Decode a key in escaped format
    --encode <encode>                Encode a key in escaped format
    --to-hex <escaped-to-hex>      Convert an escaped key to hex key
    --to-escaped <hex-to-escaped>    Convert a hex key to escaped key
    --host <host>                  Set the remote host
    --key-path <key_path>          Set the private key path
    --pd <pd>                      Set the address of pd
    --raftdb <raftdb>              Set the raft rocksdb path

SUBCOMMANDS:
    bad-regions            Get all regions with corrupt raft
    cluster                 Print the cluster id
    compact                 Compact a column family in a specified range
    compact-cluster         Compact the whole cluster in a specified range in one
                           ↪ or more column families
    consistency-check       Force a consistency-check for a specified region
    decrypt-file            Decrypt an encrypted file
    diff                    Calculate difference of region keys from different
```

```

    ↵ dbs
dump-snap-meta      Dump snapshot meta file
encryption-meta     Dump encryption metadata
fail                Inject failures to TiKV and recovery
help                Prints this message or the help of the given
    ↵ subcommand(s)
metrics              Print the metrics
modify-tikv-config  Modify tikv config, eg. tikv-ctl --host ip:port
    ↵ modify-tikv-config -n
                           rocksdb.defaultcf.disable-auto-compactions -v true
mvcc                Print the mvcc value
print               Print the raw value
raft                Print a raft log entry
raw-scan             Print all raw keys in the range
recover-mvcc        Recover mvcc data on one node by deleting corrupted
    ↵ keys
recreate-region     Recreate a region with given metadata, but alloc new
    ↵ id for it
region-properties   Show region properties
scan                Print the range db range
size                Print region size
split-region        Split the region
store               Print the store id
tombstone            Set some regions on the node to tombstone by manual
unsafe-recover      Unsafely recover the cluster when the majority
    ↵ replicas are failed

```

You can add corresponding parameters and subcommands after `tiup ctl tikv`.

11.7.1.2 General options

`tikv-ctl` provides two operation modes:

- Remote mode: use the `--host` option to accept the service address of TiKV as the argument

For this mode, if SSL is enabled in TiKV, `tikv-ctl` also needs to specify the related certificate file. For example:

```
$ tikv-ctl --ca-path ca.pem --cert-path client.pem --key-path client-
    ↵ key.pem --host 127.0.0.1:20160 <subcommands>
```

However, sometimes `tikv-ctl` communicates with PD instead of TiKV. In this case, you need to use the `--pd` option instead of `--host`. Here is an example:

```
$ tikv-ctl --pd 127.0.0.1:2379 compact-cluster
```

```
store:"127.0.0.1:20160" compact db:KV cf:default range:([], []) success
→ !
```

- Local mode: Use the `--db` option to specify the local TiKV data directory path. In this mode, you need to stop the running TiKV instance.

Unless otherwise noted, all commands support both the remote mode and the local mode.

Additionally, `tikv-ctl` has two simple commands `--to-hex` and `--to-escaped`, which are used to make simple changes to the form of the key.

Generally, use the `escaped` form of the key. For example:

```
$ tikv-ctl --to-escaped 0xaaff
\252\377
$ tikv-ctl --to-hex "\252\377"
AAFF
```

Note:

When you specify the `escaped` form of the key in a command line, it is required to enclose it in double quotes. Otherwise, bash eats the backslash and a wrong result is returned.

11.7.1.3 Subcommands, some options and flags

This section describes the subcommands that `tikv-ctl` supports in detail. Some subcommands support a lot of options. For all details, run `tikv-ctl --help <subcommand>`.

11.7.1.3.1 View information of the Raft state machine

Use the `raft` subcommand to view the status of the Raft state machine at a specific moment. The status information includes two parts: three structs (**RegionLocalState**, **RaftLocalState**, and **RegionApplyState**) and the corresponding Entries of a certain piece of log.

Use the `region` and `log` subcommands to obtain the above information respectively. The two subcommands both support the remote mode and the local mode at the same time. Their usage and output are as follows:

```
$ tikv-ctl --host 127.0.0.1:20160 raft region -r 2
region id: 2
region state key: \001\003\000\000\000\000\000\000\000\002\001
```

```

region state: Some(region {id: 2 region_epoch {conf_ver: 3 version: 1} peers
    ↪ {id: 3 store_id: 1} peers {id: 5 store_id: 4} peers {id: 7 store_id:
    ↪ 6}})
raft state key: \001\002\000\000\000\000\000\000\000\000\002\002
raft state: Some(hard_state {term: 307 vote: 5 commit: 314617} last_index:
    ↪ 314617)
apply state key: \001\002\000\000\000\000\000\000\000\000\002\003
apply state: Some(applied_index: 314617 truncated_state {index: 313474 term:
    ↪ 151})

```

11.7.1.3.2 View the Region size

Use the `size` command to view the Region size:

```
$ tikv-ctl --db /path/to/tikv/db size -r 2
region id: 2
cf default region size: 799.703 MB
cf write region size: 41.250 MB
cf lock region size: 27616
```

11.7.1.3.3 Scan to view MVCC of a specific range

The `--from` and `--to` options of the `scan` command accept two escaped forms of raw key, and use the `--show-cf` flag to specify the column families that you need to view.

```
$ tikv-ctl --db /path/to/tikv/db scan --from 'zm' --limit 2 --show-cf lock,
    ↪ default,write
key: zmBootstr\377a\377pKey
    ↪ \000\000\377\000\000\373\000\000\000\000\000\377\000\000s
    ↪ \000\000\000\000\000\372
        write cf value: start_ts: 399650102814441473 commit_ts:
            ↪ 399650102814441475 short_value: "20"
key: zmDB:29\000\000\377\000\374\000\000\000\000\000\000\377\000H
    ↪ \000\000\000\000\000\371
        write cf value: start_ts: 399650105239273474 commit_ts:
            ↪ 399650105239273475 short_value: ""
            ↪ \000\000\000\000\000\000\002"
        write cf value: start_ts: 399650105199951882 commit_ts:
            ↪ 399650105213059076 short_value: ""
            ↪ \000\000\000\000\000\000\001"
```

11.7.1.3.4 View MVCC of a given key

Similar to the `scan` command, the `mvcc` command can be used to view MVCC of a given key.

```
$ tikv-ctl --db /path/to/tikv/db mvcc -k "zmDB"
  ↳ :29\000\000\377\000\374\000\000\000\000\000\000\377\000H
  ↳ \000\000\000\000\000\000\371" --show-cf=lock,write,default
key: zmDB:29\000\000\377\000\374\000\000\000\000\000\000\377\000H
  ↳ \000\000\000\000\000\000\371
      write cf value: start_ts: 399650105239273474 commit_ts:
        ↳ 399650105239273475 short_value: "
        ↳ \000\000\000\000\000\000\002"
      write cf value: start_ts: 399650105199951882 commit_ts:
        ↳ 399650105213059076 short_value: "
        ↳ \000\000\000\000\000\000\001"
```

In this command, the key is also the escaped form of raw key.

11.7.1.3.5 Scan raw keys

The `raw-scan` command scans directly from the RocksDB. Note that to scan data keys you need to add a '`z`' prefix to keys.

Use `--from` and `--to` options to specify the range to scan (unbounded by default). Use `--limit` to limit at most how many keys to print out (30 by default). Use `--cf` to specify which cf to scan (can be `default`, `write` or `lock`).

```
$ ./tikv-ctl --db /var/lib/tikv/db/ raw-scan --from 'zt' --limit 2 --cf
  ↳ default
key: "zt\200\000\000\000\000\000\377\005_r
  ↳ \200\000\000\000\000\377\000\000\001\000\000\000\000\372\372b2
  ↳ ,^\033\377\364", value: "\010\002\002\002%\010\004\002\010root
  ↳ \010\006\002\000\010\010\t\002\010\010\n\t\002\010\014\t\002\010\016\t
  ↳ \002\010\020\t\002\010\022\t\002\010\024\t\002\010\026\t\002\010\030\
  ↳ \t\002\010\032\t\002\010\034\t\002\010\036\t\002\010\010 \t\002\010\"t
  ↳ \002\010s\t\002\010&\t\002\010(\t\002\010*\t\002\010,\t\002\010.\t
  ↳ \002\0100\t\002\0102\t\002\0104\t\002"
key: "zt\200\000\000\000\000\000\377\025_r
  ↳ \200\000\000\000\000\377\000\000\023\000\000\000\000\372\372b2
  ↳ ,^\033\377\364", value: "\010\002\002&slow_query_log_file\010\004\002
  ↳ P/usr/local/mysql/data/localhost-slow.log"
```

Total scanned keys: 2

11.7.1.3.6 Print a specific key value

To print the value of a key, use the `print` command.

11.7.1.3.7 Print some properties about Region

In order to record Region state details, TiKV writes some statistics into the SST files of Regions. To view these properties, run `tikv-ctl` with the `region-properties` sub-command:

```
$ tikv-ctl --host localhost:20160 region-properties -r 2
num_files: 0
num_entries: 0
num_deletes: 0
mvcc.min_ts: 18446744073709551615
mvcc.max_ts: 0
mvcc.num_rows: 0
mvcc.num_puts: 0
mvcc.num_versions: 0
mvcc.max_row_versions: 0
middle_key_by_approximate_size:
```

The properties can be used to check whether the Region is healthy or not. If not, you can use them to fix the Region. For example, splitting the Region manually by `middle_key_approximate_size`.

11.7.1.3.8 Compact data of each TiKV manually

Use the `compact` command to manually compact data of each TiKV. If you specify the `--from` and `--to` options, then their flags are also in the form of escaped raw key.

- Use the `--host` option to specify the TiKV that needs to perform compaction.
- Use the `-d` option to specify the RocksDB that performs compaction. The optional values are `kv` and `raft`.
- Use the `--threads` option allows you to specify the concurrency for the TiKV compaction and its default value is 8. Generally, a higher concurrency comes with a faster compaction speed, which might yet affect the service. You need to choose an appropriate concurrency count based on your scenario.
- Use the `--bottommost` option to include or exclude the bottommost files when TiKV performs compaction. The value options are `default`, `skip`, and `force`. The default value is `default`.
 - `default` means that the bottommost files are included only when the Compaction Filter feature is enabled.
 - `skip` means that the bottommost files are excluded when TiKV performs compaction.
 - `force` means that the bottommost files are always included when TiKV performs compaction.

```
$ tikv-ctl --db /path/to/tikv/db compact -d kv
success!
```

11.7.1.3.9 Compact data of the whole TiKV cluster manually

Use the `compact-cluster` command to manually compact data of the whole TiKV cluster. The flags of this command have the same meanings and usage as those of the `compact` command.

11.7.1.3.10 Set a Region to tombstone

The `tombstone` command is usually used in circumstances where the sync-log is not enabled, and some data written in the Raft state machine is lost caused by power down.

In a TiKV instance, you can use this command to set the status of some Regions to tombstone. Then when you restart the instance, those Regions are skipped to avoid the restart failure caused by damaged Raft state machines of those Regions. Those Regions need to have enough healthy replicas in other TiKV instances to be able to continue the reads and writes through the Raft mechanism.

In general cases, you can remove the corresponding Peer of this Region using the `remove ↪ -peer` command:

```
pd-ctl operator add remove-peer <region_id> <store_id>
```

Then use the `tikv-ctl` tool to set a Region to tombstone on the corresponding TiKV instance to skip the health check for this Region at startup:

```
tikv-ctl --db /path/to/tikv/db tombstone -p 127.0.0.1:2379 -r <region_id>
```

```
success!
```

However, in some cases, you cannot easily remove this Peer of this Region from PD, so you can specify the `--force` option in `tikv-ctl` to forcibly set the Peer to tombstone:

```
tikv-ctl --db /path/to/tikv/db tombstone -p 127.0.0.1:2379 -r <region_id>,<→ region_id> --force
```

```
success!
```

Note:

- The `tombstone` command only supports the local mode.
- The argument of the `-p` option specifies the PD endpoints without the `http` prefix. Specifying the PD endpoints is to query whether PD can safely switch to Tombstone.

11.7.1.3.11 Send a consistency-check request to TiKV

Use the `consistency-check` command to execute a consistency check among replicas in the corresponding Raft of a specific Region. If the check fails, TiKV itself panics. If the TiKV instance specified by `--host` is not the Region leader, an error is reported.

```
$ tikv-ctl --host 127.0.0.1:20160 consistency-check -r 2
success!
$ tikv-ctl --host 127.0.0.1:20161 consistency-check -r 2
DebugClient::check_region_consistency: RpcFailure(RpcStatus { status:
    ↪ Unknown, details: Some("StringError(\"Leader is on store 1\")") })
```

Note:

- This command only supports the remote mode.
- Even if this command returns `success!`, you need to check whether TiKV panics. This is because this command is only a proposal that requests a consistency check for the leader, and you cannot know from the client whether the whole check process is successful or not.

11.7.1.3.12 Dump snapshot meta

This sub-command is used to parse a snapshot meta file at given path and print the result.

11.7.1.3.13 Print the Regions where the Raft state machine corrupts

To avoid checking the Regions while TiKV is started, you can use the `tombstone` command to set the Regions where the Raft state machine reports an error to Tombstone. Before running this command, use the `bad-regions` command to find out the Regions with errors, so as to combine multiple tools for automated processing.

```
$ tikv-ctl --db /path/to/tikv/db bad-regions
all regions are healthy
```

If the command is successfully executed, it prints the above information. If the command fails, it prints the list of bad Regions. Currently, the errors that can be detected include the mismatches between `last index`, `commit index` and `apply index`, and the loss of Raft log. Other conditions like the damage of snapshot files still need further support.

11.7.1.3.14 View Region properties

- To view in local the properties of Region 2 on the TiKV instance that is deployed in `/path/to/tikv`:

```
$ tikv-ctl --db /path/to/tikv/data/db region-properties -r 2
```

- To view online the properties of Region 2 on the TiKV instance that is running on `127.0.0.1:20160`:

```
$ tikv-ctl --host 127.0.0.1:20160 region-properties -r 2
```

11.7.1.3.15 Modify the TiKV configuration dynamically

You can use the `modify-tikv-config` command to dynamically modify the configuration arguments. Currently, the TiKV configuration items that can be dynamically modified and the detailed modification are consistent with modifying configuration using SQL statements. For details, see [Modify TiKV configuration online](#).

- `-n` is used to specify the full name of the configuration item. For the list of configuration items that can be modified online, see [Modify TiKV configuration online](#).
- `-v` is used to specify the configuration value.

Set the size of `shared block cache`:

```
tikv-ctl --host ip:port modify-tikv-config -n storage.block-cache.capacity -  
→ v 10GB
```

```
success
```

When `shared block cache` is disabled, set `block cache size` for the `write CF`:

```
tikv-ctl --host ip:port modify-tikv-config -n rocksdb.writecf.block-cache-  
→ size -v 256MB
```

```
success
```

```
tikv-ctl --host ip:port modify-tikv-config -n raftdb.defaultcf.disable-auto-  
→ compactions -v true
```

```
success
```

```
tikv-ctl --host ip:port modify-tikv-config -n raftstore.sync-log -v false
```

```
success
```

When the compaction rate limit causes accumulated compaction pending bytes, disable the `rate-limiter-auto-tuned` mode or set a higher limit for the compaction flow:

```
tikv-ctl --host ip:port modify-tikv-config -n rocksdb.rate-limiter-auto-
↪ tuned -v false
```

success

```
tikv-ctl --host ip:port modify-tikv-config -n rocksdb.rate-bytes-per-sec -v
↪ "1GB"
```

success

11.7.1.3.16 Force Regions to recover services from failure of multiple replicas (use with caution)

You can use the `unsafe-recover remove-fail-stores` command to remove the failed machines from the peer list of Regions. Before running this command, you need to stop the service of the target TiKV store to release file locks.

The `-s` option accepts multiple `store_id` separated by comma and uses the `-r` flag to specify involved Regions. If you need to perform this operation on all Regions in a specific store, you can simply specify `--all-regions`.

Warning:

- If any misoperation is performed, it might be hard to recover the cluster. Be aware of the potential risks and avoid using this feature in a production environment.
- If the `--all-regions` option is used, you are expected to run this command on all the remaining stores connected to the cluster. You need to ensure that these healthy stores stop providing services before recovering the damaged stores. Otherwise, the inconsistent peer lists in Region replicas will cause errors when you run `split-region` or `remove-peer`. This further causes inconsistency between other metadata, and finally, the Regions will become unavailable.
- Once you have run `remove-fail-stores`, you cannot restart the removed nodes or add these nodes to the cluster. Otherwise, the metadata will be inconsistent, and finally, the Regions will be unavailable.

```
tikv-ctl --db /path/to/tikv/db unsafe-recover remove-fail-stores -s 3 -r
↪ 1001,1002
```

```
success!
```

```
tikv-ctl --db /path/to/tikv/db unsafe-recover remove-fail-stores -s 4,5 --
    ↪ all-regions
```

Then, after you restart TiKV, the Regions can continue providing services with the remaining healthy replicas. This command is commonly used when multiple TiKV stores are damaged or deleted.

Note:

- You are expected to run this command for all stores where the specified Regions' peers are located.
- This command only supports the local mode. It prints `success!` when successfully run.

11.7.1.3.17 Recover from MVCC data corruption

Use the `recover-mvcc` command in circumstances where TiKV cannot run normally caused by MVCC data corruption. It cross-checks 3 CFs (“default”, “write”, “lock”) to recover from various kinds of inconsistency.

- Use the `-r` option to specify involved Regions by `region_id`.
- Use the `-p` option to specify PD endpoints.

```
$ tikv-ctl --db /path/to/tikv/db recover-mvcc -r 1001,1002 -p 127.0.0.1:2379
success!
```

Note:

- This command only supports the local mode. It prints `success!` when successfully run.
- The argument of the `-p` option specifies the PD endpoints without the `http` prefix. Specifying the PD endpoints is to query whether the specified `region_id` is validated or not.
- You need to run this command for all stores where specified Regions' peers are located.

11.7.1.3.18 Ldb Command

The `ldb` command line tool offers multiple data access and database administration commands. Some examples are listed below. For more information, refer to the help message displayed when running `tikv-ctl ldb` or check the documents from RocksDB.

Examples of data access sequence:

To dump an existing RocksDB in HEX:

```
$ tikv-ctl ldb --hex --db=/tmp/db dump
```

To dump the manifest of an existing RocksDB:

```
$ tikv-ctl ldb --hex manifest_dump --path=/tmp/db/MANIFEST-000001
```

You can specify the column family that your query is against using the `--column_family` → `=<string>` command line.

`--try_load_options` loads the database options file to open the database. It is recommended to always keep this option on when the database is running. If you open the database with default options, the LSM-tree might be messed up, which cannot be recovered automatically.

11.7.1.3.19 Dump encryption metadata

Use the `encryption-meta` subcommand to dump encryption metadata. The subcommand can dump two types of metadata: encryption info for data files, and the list of data encryption keys used.

To dump encryption info for data files, use the `encryption-meta dump-file` subcommand. You need to create a TiKV config file to specify `data-dir` for the TiKV deployment:

```
### conf.toml
[storage]
data-dir = "/path/to/tikv/data"
```

The `--path` option can be used to specify an absolute or relative path to the data file of interest. The command might give empty output if the data file is not encrypted. If `--path` is not provided, encryption info for all data files will be printed.

```
$ tikv-ctl --config=./conf.toml encryption-meta dump-file --path=/path/to/
  ↪ tikv/data/db/CURRENT
/path/to/tikv/data/db/CURRENT: key_id: 9291156302549018620 iv:
  ↪ E3C2FDBF63FC03BFC28F265D7E78283F method: Aes128Ctr
```

To dump data encryption keys, use the `encryption-meta dump-key` subcommand. In addition to `data-dir`, you also need to specify the current master key used in the config file. For how to config master key, refer to [Encryption-At-Rest](#). Also with this command, the `security.encryption.previous-master-key` config will be ignored, and the master key rotation will not be triggered.

```
### conf.toml
[storage]
data-dir = "/path/to/tikv/data"

[security.encryption.master-key]
type = "kms"
key-id = "0987dcba-09fe-87dc-65ba-ab0987654321"
region = "us-west-2"
```

Note if the master key is a AWS KMS key, `tikv-ctl` needs to have access to the KMS key. Access to a AWS KMS key can be granted to `tikv-ctl` via environment variable, AWS default config file, or IAM role, whichever is suitable. Refer to AWS document for usage.

The `--ids` option can be used to specified a list of comma-separated data encryption key ids to print. If `--ids` is not provided, all data encryption keys will be printed, along with current key id, which is the id of the latest active data encryption key.

When using the command, you will see a prompt warning that the action will expose sensitive information. Type “I consent” to continue.

```
$ ./tikv-ctl --config=./conf.toml encryption-meta dump-key
This action will expose encryption key(s) as plaintext. Do not output the
↪ result in file on disk.
Type "I consent" to continue, anything else to exit: I consent
current key id: 9291156302549018620
9291156302549018620: key: 8B6B6B8F83D36BE2467ED55D72AE808B method: Aes128Ctr
↪ creation_time: 1592938357
```

```
$ ./tikv-ctl --config=./conf.toml encryption-meta dump-key --ids
↪ =9291156302549018620
This action will expose encryption key(s) as plaintext. Do not output the
↪ result in file on disk.
Type "I consent" to continue, anything else to exit: I consent
9291156302549018620: key: 8B6B6B8F83D36BE2467ED55D72AE808B method: Aes128Ctr
↪ creation_time: 1592938357
```

Note

The command will expose data encryption keys as plaintext. In production, DO NOT redirect the output to a file. Even deleting the output file afterward may not cleanly wipe out the content from disk.

11.7.1.3.20 Print information related to damaged SST files

Damaged SST files in TiKV might cause the TiKV process to panic. To clean up the damaged SST files, you will need the information of these files. To get the information, you can execute the `bad-ssts` command in TiKV Control. The needed information is shown in the output. The following is an example command and output.

Note:

Before running this command, stop the running TiKV instance.

```
$ tikv-ctl bad-ssts --db </path/to/tikv/db> --pd <endpoint>
```

```
-----
corruption info:
data/tikv-21107/db/000014.sst: Corruption: Bad table magic number: expected
    ↪ 9863518390377041911, found 759105309091689679 in data/tikv-21107/db
    ↪ /000014.sst

sst meta:
14:552997[1 .. 5520]['0101' seq:1, type:1 .. '7
    ↪ A74800000000000FF0F5F728000000000FF0002160000000000FAFA13AB33020BFFF
    ↪ ' seq:2032, type:1] at level 0 for Column family "default" (ID 0)
it isn't easy to handle local data, start key:0101

overlap region:
RegionInfo { region: id: 4 end_key: 748000000000000FF0500000000000000000000F8
    ↪ region_epoch { conf_ver: 1 version: 2 } peers { id: 5 store_id: 1 },
    ↪ leader: Some(id: 5 store_id: 1) }

suggested operations:
tikv-ctl ldb --db=data/tikv-21107/db unsafe_remove_sst_file "data/tikv
    ↪ -21107/db/000014.sst"
tikv-ctl --db=data/tikv-21107/db tombstone -r 4 --pd <endpoint>
-----
corruption analysis has completed
```

From the output above, you can see that the information of the damaged SST file is printed first and then the meta-information is printed.

- In the `sst meta` part, 14 means the SST file number; 552997 means the file size, followed by the smallest and largest sequence numbers and other meta-information.

- The `overlap region` part shows the information of the Region involved. This information is obtained through the PD server.
- The `suggested operations` part provides you suggestion to clean up the damaged SST file. You can take the suggestion to clean up files and restart the TiKV instance.

11.7.2 PD Control User Guide

As a command line tool of PD, PD Control obtains the state information of the cluster and tunes the cluster.

11.7.2.1 Install PD Control

Note:

It is recommended that the version of the Control tool you use is consistent with the version of the cluster.

11.7.2.1.1 Use TiUP command

To use PD Control, execute the `tiup ctl:<cluster-version> pd -u http://<pd_ip>:<pd_port> [-i]` command.

11.7.2.1.2 Download TiDB installation package

If you want to download the latest version of `pd-ctl`, directly download the TiDB package, because `pd-ctl` is included in the TiDB package.

Package				
down-				SHA256
load				check-
link	OS	Architecture	Summe	
https://download.pingcap.org/	Linux	amd64	https://download.pingcap.org/	
↳ ://			↳ ://	
↳ download			↳ download	
↳ .			↳ .	
↳ pingcap			↳ pingcap	
↳ .			↳ .	
↳ org			↳ org	
↳ /			↳ /	
↳ tidb			↳ tidb	
↳ -{			↳ -{	
↳ version			↳ version	
↳ }-			↳ }-	
↳ linux			↳ linux	
↳ -			↳ -	
↳ amd64			↳ amd64	
↳ .			↳ .	
↳ tar			↳ sha256	
↳ .			↳	
↳ gz				
↳				
(pd- ctl)				

Note:

{version} indicates the version number of TiDB. For example, if {version} is v5.3.0, the package download link is <https://download.pingcap.org/tidb-v5.3.0-linux-amd64.tar.gz>.

11.7.2.1.3 Compile from source code

1. Go Version 1.13 or later because the Go modules are used.
2. In the root directory of the PD project, use the `make` or `make pd-ctl` command to compile and generate `bin/pd-ctl`.

11.7.2.2 Usage

Single-command mode:

```
tiup ctl pd store -u http://127.0.0.1:2379
```

Interactive mode:

```
tiup ctl pd -i -u http://127.0.0.1:2379
```

Use environment variables:

```
export PD_ADDR=http://127.0.0.1:2379  
tiup ctl pd
```

Use TLS to encrypt:

```
tiup ctl pd -u https://127.0.0.1:2379 --cacert="path/to/ca" --cert="path/to/  
↪ cert" --key="path/to/key"
```

11.7.2.3 Command line flags

11.7.2.3.1 --cacert

- Specifies the path to the certificate file of the trusted CA in PEM format
- Default: “”

11.7.2.3.2 --cert

- Specifies the path to the certificate of SSL in PEM format
- Default: “”

11.7.2.3.3 --detach / -d

- Uses the single command line mode (not entering readline)
- Default: true

11.7.2.3.4 --help / -h

- Outputs the help information
- Default: false

11.7.2.3.5 --interact / -i

- Uses the interactive mode (entering readline)
- Default: false

11.7.2.3.6 --key

- Specifies the path to the certificate key file of SSL in PEM format, which is the private key of the certificate specified by --cert
- Default: “”

11.7.2.3.7 --pd / -u

- Specifies the PD address
- Default address: `http://127.0.0.1:2379`
- Environment variable: `PD_ADDR`

11.7.2.3.8 --version / -V

- Prints the version information and exit
- Default: false

11.7.2.4 Command

11.7.2.4.1 cluster

Use this command to view the basic information of the cluster.

Usage:

```
>> cluster                                // To show the cluster information
{
  "id": 6493707687106161130,
  "max_peer_count": 3
}
```

11.7.2.4.2 config [show | set <option> <value> | placement-rules]

Use this command to view or modify the configuration information.

Usage:

```
>> config show                            // Display the config information
  ↳ of the scheduling
{
  "replication": {
    "enable-placement-rules": "true",
    "isolation-level": "",
    "location-labels": "",
    "max-replicas": 3,
```

```

    "strictly-match-label": "false"
},
"schedule": {
    "enable-cross-table-merge": "true",
    "high-space-ratio": 0.7,
    "hot-region-cache-hits-threshold": 3,
    "hot-region-schedule-limit": 4,
    "leader-schedule-limit": 4,
    "leader-schedule-policy": "count",
    "low-space-ratio": 0.8,
    "max-merge-region-keys": 200000,
    "max-merge-region-size": 20,
    "max-pending-peer-count": 64,
    "max-snapshot-count": 64,
    "max-store-down-time": "30m0s",
    "merge-schedule-limit": 8,
    "patrol-region-interval": "10ms",
    "region-schedule-limit": 2048,
    "region-score-formula-version": "v2",
    "replica-schedule-limit": 64,
    "scheduler-max-waiting-operator": 5,
    "split-merge-interval": "1h0m0s",
    "tolerant-size-ratio": 0
}
}
>> config show all                      // Display all config information
>> config show replication             // Display the config information
   ↪ of replication
{
    "max-replicas": 3,
    "location-labels": "",
    "isolation-level": "",
    "strictly-match-label": "false",
    "enable-placement-rules": "true"
}

>> config show cluster-version          // Display the current version of
   ↪ the cluster, which is the current minimum version of TiKV nodes in
   ↪ the cluster and does not correspond to the binary version.
"5.2.2"

```

- `max-snapshot-count` controls the maximum number of snapshots that a single store receives or sends out at the same time. The scheduler is restricted by this configuration to avoid taking up normal application resources. When you need to improve the speed

of adding replicas or balancing, increase this value.

```
>> config set max-snapshot-count 64 // Set the maximum number of
   ↪ snapshots to 64
```

- **max-pending-peer-count** controls the maximum number of pending peers in a single store. The scheduler is restricted by this configuration to avoid producing a large number of Regions without the latest log in some nodes. When you need to improve the speed of adding replicas or balancing, increase this value. Setting it to 0 indicates no limit.

```
>> config set max-pending-peer-count 64 // Set the maximum number of
   ↪ pending peers to 64
```

- **max-merge-region-size** controls the upper limit on the size of Region Merge (the unit is M). When `regionSize` exceeds the specified value, PD does not merge it with the adjacent Region. Setting it to 0 indicates disabling Region Merge.

```
>> config set max-merge-region-size 16 // Set the upper limit on the
   ↪ size of Region Merge to 16M
```

- **max-merge-region-keys** controls the upper limit on the key count of Region Merge. When `regionKeyCount` exceeds the specified value, PD does not merge it with the adjacent Region.

```
>> config set max-merge-region-keys 50000 // Set the the upper limit on
   ↪ keyCount to 50000
```

- **split-merge-interval** controls the interval between the `split` and `merge` operations on a same Region. This means the newly split Region won't be merged within a period of time.

```
>> config set split-merge-interval 24h // Set the interval between `split` and `merge` to one day
```

- **enable-one-way-merge** controls whether PD only allows a Region to merge with the next Region. When you set it to `false`, PD allows a Region to merge with the adjacent two Regions.

```
>> config set enable-one-way-merge true // Enables one-way merging.
```

- **enable-cross-table-merge** is used to enable the merging of cross-table Regions. When you set it to `false`, PD does not merge the Regions from different tables. This option only works when key type is "table".

```
>> config set enable-cross-table-merge true // Enable cross table merge
   ↪ .
```

- **key-type** specifies the key encoding type used for the cluster. The supported options are [“table”, “raw”, “txn”], and the default value is “table”.
 - If no TiDB instance exists in the cluster, **key-type** will be “raw” or “txn”, and PD is allowed to merge Regions across tables regardless of the `enable-cross-table-merge` setting.
 - If any TiDB instance exists in the cluster, **key-type** should be “table”. Whether PD can merge Regions across tables is determined by `enable-cross-table-merge`. If **key-type** is “raw”, placement rules do not work.

```
>> config set key-type raw // Enable cross table merge.
```

- **region-score-formula-version** controls the version of the Region score formula. The value options are v1 and v2. The version 2 of the formula helps to reduce redundant balance Region scheduling in some scenarios, such as taking TiKV nodes online or offline.

```
>> config set region-score-formula-version v2
```

- **patrol-region-interval** controls the execution frequency that `replicaChecker` checks the health status of Regions. A shorter interval indicates a higher execution frequency. Generally, you do not need to adjust it.

```
>> config set patrol-region-interval 10ms // Set the execution
    ↪ frequency of replicaChecker to 10ms
```

- **max-store-down-time** controls the time that PD decides the disconnected store cannot be restored if exceeded. If PD does not receive heartbeats from a store within the specified period of time, PD adds replicas in other nodes.

```
>> config set max-store-down-time 30m // Set the time within which PD
    ↪ receives no heartbeats and after which PD starts to add replicas
    ↪ to 30 minutes
```

- **leader-schedule-limit** controls the number of tasks scheduling the leader at the same time. This value affects the speed of leader balance. A larger value means a higher speed and setting the value to 0 closes the scheduling. Usually the leader scheduling has a small load, and you can increase the value in need.

```
>> config set leader-schedule-limit 4 // 4 tasks of leader scheduling
    ↪ at the same time at most
```

- **region-schedule-limit** controls the number of tasks of scheduling Regions at the same time. This value avoids too many Region balance operators being created. The default value is 2048 which is enough for all sizes of clusters, and setting the value to 0

closes the scheduling. Usually, the Region scheduling speed is limited by `store-limit`, but it is recommended that you do not customize this value unless you know exactly what you are doing.

```
>> config set region-schedule-limit 2 // 2 tasks of Region scheduling
    ↪ at the same time at most
```

- `replica-schedule-limit` controls the number of tasks scheduling the replica at the same time. This value affects the scheduling speed when the node is down or removed. A larger value means a higher speed and setting the value to 0 closes the scheduling. Usually the replica scheduling has a large load, so do not set a too large value.

```
>> config set replica-schedule-limit 4 // 4 tasks of replica
    ↪ scheduling at the same time at most
```

- `merge-schedule-limit` controls the number of Region Merge scheduling tasks. Setting the value to 0 closes Region Merge. Usually the Merge scheduling has a large load, so do not set a too large value.

```
>> config set merge-schedule-limit 16 // 16 tasks of Merge scheduling
    ↪ at the same time at most
```

- `hot-region-schedule-limit` controls the hot Region scheduling tasks that are running at the same time. Setting its value to 0 means to disable the scheduling. It is not recommended to set a too large value, otherwise it might affect the system performance.

```
>> config set hot-region-schedule-limit 4 // 4 tasks of hot Region
    ↪ scheduling at the same time at most
```

- `hot-region-cache-hits-threshold` is used to set the number of minutes required to identify a hot Region. PD can participate in the hotspot scheduling only after the Region is in the hotspot state for more than this number of minutes.

- `tolerant-size-ratio` controls the size of the balance buffer area. When the score difference between the leader or Region of the two stores is less than specified multiple times of the Region size, it is considered in balance by PD.

```
>> config set tolerant-size-ratio 20 // Set the size of the buffer
    ↪ area to about 20 times of the average Region Size
```

- `low-space-ratio` controls the threshold value that is considered as insufficient store space. When the ratio of the space occupied by the node exceeds the specified value, PD tries to avoid migrating data to the corresponding node as much as possible. At the same time, PD mainly schedules the remaining space to avoid using up the disk space of the corresponding node.

```
config set low-space-ratio 0.9          // Set the threshold value of
→ insufficient space to 0.9
```

- `high-space-ratio` controls the threshold value that is considered as sufficient store space. This configuration takes effect only when `region-score-formula-version` is set to v1. When the ratio of the space occupied by the node is less than the specified value, PD ignores the remaining space and mainly schedules the actual data volume.

```
config set high-space-ratio 0.5        // Set the threshold value of
→ sufficient space to 0.5
```

- `cluster-version` is the version of the cluster, which is used to enable or disable some features and to deal with the compatibility issues. By default, it is the minimum version of all normally running TiKV nodes in the cluster. You can set it manually only when you need to roll it back to an earlier version.

```
config set cluster-version 1.0.8       // Set the version of the
→ cluster to 1.0.8
```

- `replication-mode` controls the replication mode of Regions in the dual data center scenario. See [Enable the DR Auto-Sync mode](#) for details.
- `leader-schedule-policy` is used to select the scheduling strategy for the leader. You can schedule the leader according to `size` or `count`.
- `scheduler-max-waiting-operator` is used to control the number of waiting operators in each scheduler.
- `enable-remove-down-replica` is used to enable the feature of automatically deleting DownReplica. When you set it to `false`, PD does not automatically clean up the downtime replicas.
- `enable-replace-offline-replica` is used to enable the feature of migrating OfflineReplica. When you set it to `false`, PD does not migrate the offline replicas.
- `enable-make-up-replica` is used to enable the feature of making up replicas. When you set it to `false`, PD does not add replicas for Regions without sufficient replicas.
- `enable-remove-extra-replica` is used to enable the feature of removing extra replicas. When you set it to `false`, PD does not remove extra replicas for Regions with redundant replicas.
- `enable-location-replacement` is used to enable the isolation level checking. When you set it to `false`, PD does not increase the isolation level of a Region replica through scheduling.

- `enable-debug-metrics` is used to enable the metrics for debugging. When you set it to `true`, PD enables some metrics such as `balance-tolerant-size`.
- `enable-placement-rules` is used to enable placement rules, which is enabled by default in v5.0 and later versions.
- `store-limit-mode` is used to control the mode of limiting the store speed. The optional modes are `auto` and `manual`. In `auto` mode, the stores are automatically balanced according to the load (experimental).
- PD rounds the lowest digits of the flow number, which reduces the update of statistics caused by the changes of the Region flow information. This configuration item is used to specify the number of lowest digits to round for the Region flow information. For example, the flow 100512 will be rounded to 101000 because the default value is 3. This configuration replaces `trace-region-flow`.
- For example, set the value of `flow-round-by-digit` to 4:

```
config set flow-round-by-digit 4
```

`config placement-rules [disable | enable | load | save | show | rule-group]`

For the usage of `config placement-rules [disable | enable | load | save | show | rule-group]`, see [Configure placement rules](#).

11.7.2.4.3 health

Use this command to view the health information of the cluster.

Usage:

```
>> health                                // Display the health information
[
  {
    "name": "pd",
    "member_id": 13195394291058371180,
    "client_urls": [
      "http://127.0.0.1:2379"
      .....
    ],
    "health": true
  }
  .....
]
```

11.7.2.4.4 hot [read | write | store]

Use this command to view the hot spot information of the cluster.

Usage:

```
>> hot read                      // Display hot spot for the read
    ↵ operation
>> hot write                     // Display hot spot for the write
    ↵ operation
>> hot store                     // Display hot spot for all the read and
    ↵ write operations
```

11.7.2.4.5 label [store <name> <value>]

Use this command to view the label information of the cluster.

Usage:

```
>> label                         // Display all labels
>> label store zone cn          // Display all stores including the "zone"
    ↵ ":"cn" label
```

11.7.2.4.6 member [delete | leader_priority | leader [show | resign | transfer <member_name>]]

Use this command to view the PD members, remove a specified member, or configure the priority of leader.

Usage:

```
>> member                         // Display the information of all members
{
  "header": {.....},
  "members": [.....],
  "leader": {.....},
  "etcd_leader": {.....},
}
>> member delete name pd2        // Delete "pd2"
Success!
>> member delete id 1319539429105371180 // Delete a node using id
Success!
>> member leader show           // Display the leader information
{
  "name": "pd",
  "member_id": 13155432540099656863,
  "peer_urls": [.....],
  "client_urls": [.....]
```

```

}
>> member leader resign // Move leader away from the current member
.....
>> member leader transfer pd3 // Migrate leader to a specified member
.....

```

11.7.2.4.7 operator [check | show | add | remove]

Use this command to view and control the scheduling operation.

Usage:

>> operator show	// Display all operators
>> operator show admin	// Display all admin
↳ operators	
>> operator show leader	// Display all leader
↳ operators	
>> operator show region	// Display all Region
↳ operators	
>> operator add add-peer 1 2	// Add a replica of Region
↳ 1 on store 2	
>> operator add add-learner 1 2	// Add a learner replica of
↳ Region 1 on store 2	
>> operator add remove-peer 1 2	// Remove a replica of
↳ Region 1 on store 2	
>> operator add transfer-leader 1 2	// Schedule the leader of
↳ Region 1 to store 2	
>> operator add transfer-region 1 2 3 4	// Schedule Region 1 to
↳ stores 2,3,4	
>> operator add transfer-peer 1 2 3	// Schedule the replica of
↳ Region 1 on store 2 to store 3	
>> operator add merge-region 1 2	// Merge Region 1 with
↳ Region 2	
>> operator add split-region 1 --policy=approximate	// Split Region 1 into
↳ two Regions in halves, based on approximately estimated value	
>> operator add split-region 1 --policy=scan	// Split Region 1 into two
↳ Regions in halves, based on accurate scan value	
>> operator remove 1	// Remove the scheduling
↳ operation of Region 1	
>> operator check 1	// Check the status of the
↳ operators related to Region 1	

The splitting of Regions starts from the position as close as possible to the middle. You can locate this position using two strategies, namely “scan” and “approximate”. The difference between them is that the former determines the middle key by scanning the Region,

and the latter obtains the approximate position by checking the statistics recorded in the SST file. Generally, the former is more accurate, while the latter consumes less I/O and can be completed faster.

11.7.2.4.8 ping

Use this command to view the time that ping PD takes.

Usage:

```
>> ping
time: 43.12698ms
```

11.7.2.4.9 region <region_id> [--jq=""]

Use this command to view the Region information. For a jq formatted output, see [jq-formatted-json-output-usage](#).

Usage:

```
>> region                                // Display the information of all
    ↪ Regions
{
  "count": 1,
  "regions": [.....]
}

>> region 2                               // Display the information of the Region
    ↪ with the ID of 2
{
  "id": 2,
  "start_key": "7480000000000000FF1D000000000000F8",
  "end_key": "7480000000000000FF1F000000000000F8",
  "epoch": {
    "conf_ver": 1,
    "version": 15
  },
  "peers": [
    {
      "id": 40,
      "store_id": 3
    }
  ],
  "leader": {
    "id": 40,
    "store_id": 3
  },
}
```

```

    "written_bytes": 0,
    "read_bytes": 0,
    "written_keys": 0,
    "read_keys": 0,
    "approximate_size": 1,
    "approximate_keys": 0
}

```

11.7.2.4.10 region key [--format=raw|encode|hex] <key>

Use this command to query the Region that a specific key resides in. It supports the raw, encoding, and hex formats. And you need to use single quotes around the key when it is in the encoding format.

Hex format usage (default):

```

>> region key 7480000000000000FF1300000000000000F8
{
  "region": {
    "id": 2,
    .....
  }
}

```

Raw format usage:

```

>> region key --format=raw abc
{
  "region": {
    "id": 2,
    .....
  }
}

```

Encoding format usage:

```

>> region key --format=encode 't\200\000\000\000\000\000\000\377\035_r
  ↵ \200\000\000\000\377\017U\320\000\000\000\000\372'
{
  "region": {
    "id": 2,
    .....
  }
}

```

11.7.2.4.11 region scan

Use this command to get all Regions.

Usage:

```
>> region scan
{
  "count": 20,
  "regions": [.....],
}
```

11.7.2.4.12 region sibling <region_id>

Use this command to check the adjacent Regions of a specific Region.

Usage:

```
>> region sibling 2
{
  "count": 2,
  "regions": [.....],
}
```

11.7.2.4.13 region startkey [--format=raw|encode|hex] <key> <limit>

Use this command to query all Regions starting from a key.

Usage:

```
>> region startkey --format=raw abc
{
  "count": 16,
  "regions": [.....],
}
```

11.7.2.4.14 region store <store_id>

Use this command to list all Regions of a specific store.

Usage:

```
>> region store 2
{
  "count": 10,
  "regions": [.....],
}
```

11.7.2.4.15 region topread [limit]

Use this command to list Regions with top read flow. The default value of the limit is 16.

Usage:

```
>> region topread
{
  "count": 16,
  "regions": [.....],
}
```

11.7.2.4.16 region topwrite [limit]

Use this command to list Regions with top write flow. The default value of the limit is 16.

Usage:

```
>> region topwrite
{
  "count": 16,
  "regions": [.....],
}
```

11.7.2.4.17 region topconfver [limit]

Use this command to list Regions with top conf version. The default value of the limit is 16.

Usage:

```
>> region topconfver
{
  "count": 16,
  "regions": [.....],
}
```

11.7.2.4.18 region topversion [limit]

Use this command to list Regions with top version. The default value of the limit is 16.

Usage:

```
>> region topversion
{
  "count": 16,
  "regions": [.....],
```

}

11.7.2.4.19 region topsize [limit]

Use this command to list Regions with top approximate size. The default value of the limit is 16.

Usage:

```
>> region topsize
{
  "count": 16,
  "regions": [.....],
}
```

11.7.2.4.20 region check [miss-peer | extra-peer | down-peer | pending-peer | offline-peer | empty-region | hist-size | hist-keys]

Use this command to check the Regions in abnormal conditions.

Description of various types:

- miss-peer: the Region without enough replicas
- extra-peer: the Region with extra replicas
- down-peer: the Region in which some replicas are Down
- pending-peer: the Region in which some replicas are Pending

Usage:

```
>> region check miss-peer
{
  "count": 2,
  "regions": [.....],
}
```

11.7.2.4.21 scheduler [show | add | remove | pause | resume | config]

Use this command to view and control the scheduling policy.

Usage:

```
>> scheduler show                      // Display all schedulers
>> scheduler add grant-leader-scheduler 1 // Schedule all the leaders of
   ↪ the Regions on store 1 to store 1
>> scheduler add evict-leader-scheduler 1 // Move all the Region leaders
   ↪ on store 1 out
```

```

>> scheduler config evict-leader-scheduler // Display the stores in which
    ↪ the scheduler is located since v4.0.0
>> scheduler add shuffle-leader-scheduler // Randomly exchange the leader
    ↪ on different stores
>> scheduler add shuffle-region-scheduler // Randomly scheduling the
    ↪ Regions on different stores
>> scheduler add evict-slow-store-scheduler // When there is one and only
    ↪ one slow store, evict all Region leaders of that store
>> scheduler remove grant-leader-scheduler-1 // Remove the corresponding
    ↪ scheduler, and ` -1` corresponds to the store ID
>> scheduler pause balance-region-scheduler 10 // Pause the balance-region
    ↪ scheduler for 10 seconds
>> scheduler pause all 10                      // Pause all schedulers for 10
    ↪ seconds
>> scheduler resume balance-region-scheduler // Continue to run the balance-
    ↪ region scheduler
>> scheduler resume all                      // Continue to run all
    ↪ schedulers
>> scheduler config balance-hot-region-scheduler // Display the
    ↪ configuration of the balance-hot-region scheduler

```

`scheduler config balance-hot-region-scheduler`

Use this command to view and control the `balance-hot-region-scheduler` policy.

Usage:

```

>> scheduler config balance-hot-region-scheduler // Display all
    ↪ configuration of the balance-hot-region scheduler
{
  "min-hot-byte-rate": 100,
  "min-hot-key-rate": 10,
  "min-hot-query-rate": 10,
  "max-zombie-rounds": 3,
  "max-peer-number": 1000,
  "byte-rate-rank-step-ratio": 0.05,
  "key-rate-rank-step-ratio": 0.05,
  "query-rate-rank-step-ratio": 0.05,
  "count-rank-step-ratio": 0.01,
  "great-dec-ratio": 0.95,
  "minor-dec-ratio": 0.99,
  "src-tolerance-ratio": 1.05,
  "dst-tolerance-ratio": 1.05,
  "read-priorities": [
    "query",
    "byte"
  ],
}

```

```

"write-leader-priorities": [
    "key",
    "byte"
],
"write-peer-priorities": [
    "byte",
    "key"
],
"strict-picking-store": "true",
"enable-for-tiflash": "true"
}

```

- **min-hot-byte-rate** means the smallest number of bytes to be counted, which is usually 100.

```

>> scheduler config balance-hot-region-scheduler set min-hot-byte-rate
    ↵ 100

```

- **min-hot-key-rate** means the smallest number of keys to be counted, which is usually 10.

```

>> scheduler config balance-hot-region-scheduler set min-hot-key-rate
    ↵ 10

```

- **min-hot-query-rate** means the smallest number of queries to be counted, which is usually 10.

```

>> scheduler config balance-hot-region-scheduler set min-hot-query-rate
    ↵ 10

```

- **max-zombie-rounds** means the maximum number of heartbeats with which an operator can be considered as the pending influence. If you set it to a larger value, more operators might be included in the pending influence. Usually, you do not need to adjust its value. Pending influence refers to the operator influence that is generated during scheduling but still has an effect.

```

>> scheduler config balance-hot-region-scheduler set max-zombie-rounds
    ↵ 3

```

- **max-peer-number** means the maximum number of peers to be solved, which prevents the scheduler from being too slow.

```

>> scheduler config balance-hot-region-scheduler set max-peer-number
    ↵ 1000

```

- `byte-rate-rank-step-ratio`, `key-rate-rank-step-ratio`, `query-rate-rank-step-ratio`, and `count-rank-step-ratio` respectively mean the step ranks of byte, key, query, and count. The rank-step-ratio decides the step when the rank is calculated. `great-dec-ratio` and `minor-dec-ratio` are used to determine the dec rank. Usually, you do not need to modify these items.

```
>> scheduler config balance-hot-region-scheduler set byte-rate-rank-
    ↪ step-ratio 0.05
```

- `src-tolerance-ratio` and `dst-tolerance-ratio` are configuration items for the expectation scheduler. The smaller the `tolerance-ratio`, the easier it is for scheduling. When redundant scheduling occurs, you can appropriately increase this value.

```
>> scheduler config balance-hot-region-scheduler set src-tolerance-
    ↪ ratio 1.1
```

- `read-priorities`, `write-leader-priorities`, and `write-peer-priorities` control which dimension the scheduler prioritizes for hot Region scheduling. Two dimensions are supported for configuration.
 - `read-priorities` and `write-leader-priorities` control which dimensions the scheduler prioritizes for scheduling hot Regions of the read and write-leader types. The dimension options are `query`, `byte`, and `key`.
 - `write-peer-priorities` controls which dimensions the scheduler prioritizes for scheduling hot Regions of the write-peer type. The dimension options are `byte` and `key`.

Note:

If a cluster component is earlier than v5.2, the configuration of `query` dimension does not take effect. If some components are upgraded to v5.2 or later, the `byte` and `key` dimensions still by default have the priority for hot Region scheduling. After all components of the cluster are upgraded to v5.2 or later, such a configuration still takes effect for compatibility. You can view the real-time configuration using the `pd-ctl` command. Usually, you do not need to modify these configurations.

```
>> scheduler config balance-hot-region-scheduler set read-priorities
    ↪ query,byte
```

- `strict-picking-store` controls the search space of hot Region scheduling. Usually, it is enabled. When it is enabled, hot Region scheduling ensures hotspot balance on the two configured dimensions. When it is disabled, hot Region scheduling only ensures the balance on the dimension with the first priority, which might reduce balance on other dimensions. Usually, you do not need to modify this configuration.

```
>> scheduler config balance-hot-region-scheduler set strict-picking-
    ↪ store true
```

- enable-for-tiflash controls whether hot Region scheduling takes effect for TiFlash instances. Usually, it is enabled. When it is disabled, the hot Region scheduling between TiFlash instances is not performed.

```
>> scheduler config balance-hot-region-scheduler set enable-for-tiflash
    ↪ true
```

11.7.2.4.22 store [delete | label | weight | remove-tombstone | limit] <store_id> [--jq="*query string*"]

Use this command to view the store information or remove a specified store. For a jq formatted output, see [jq-formatted-json-output-usage](#).

Usage:

```
>> store                                // Display information of all stores
{
  "count": 3,
  "stores": [...]
}
>> store 1                               // Get the store with the store id of 1
.....
>> store delete 1                         // Delete the store with the store id of
    ↪ 1
.....
>> store label 1 zone cn                // Set the value of the label with the "z
    ↪ one" key to "cn" for the store with the store id of 1
>> store weight 1 5 10                  // Set the leader weight to 5 and Region
    ↪ weight to 10 for the store with the store id of 1
>> store remove-tombstone             // Remove stores that are in tombstone
    ↪ state
>> store limit                          // Show the speed limit of adding-peer
    ↪ operations and the limit of removing-peer operations per minute in
    ↪ all stores
>> store limit add-peer                // Show the speed limit of adding-peer
    ↪ operations per minute in all stores
>> store limit remove-peer            // Show the limit of removing-peer
    ↪ operations per minute in all stores
>> store limit all 5                  // Set the limit of adding-peer
    ↪ operations to 5 and the limit of removing-peer operations to 5 per
    ↪ minute for all stores
```

```

>> store limit 1 5          // Set the limit of adding-peer
    ↳ operations to 5 and the limit of removing-peer operations to 5 per
    ↳ minute for store 1
>> store limit all 5 add-peer // Set the limit of adding-peer
    ↳ operations to 5 per minute for all stores
>> store limit 1 5 add-peer // Set the limit of adding-peer
    ↳ operations to 5 per minute for store 1
>> store limit 1 5 remove-peer // Set the limit of removing-peer
    ↳ operations to 5 per minute for store 1
>> store limit all 5 remove-peer // Set the limit of removing-peer
    ↳ operations to 5 per minute for all stores

```

Note:

When you use the `store limit` command, the original `region-add` and `region-remove` are deprecated. Use `add-peer` and `remove-peer` instead.

11.7.2.4.23 log [fatal | error | warn | info | debug]

Use this command to set the log level of the PD leader.

Usage:

```
>> log warn
```

11.7.2.4.24 tso

Use this command to parse the physical and logical time of TSO.

Usage:

```

>> tso 395181938313123110 // Parse TSO
system: 2017-10-09 05:50:59 +0800 CST
logic: 120102

```

11.7.2.4.25 unsafe remove-failed-stores [store-ids | show | history]

Warning:

- This feature is a lossy recovery, so TiKV cannot guarantee data integrity and data indexes integrity after using the feature.

- Online Unsafe Recovery is an experimental feature, and it is **NOT** recommended to use it in the production environment. The interface, strategy, and internal implementation of this feature might change when it becomes generally available (GA). Although this feature has been tested in some scenarios, it is not thoroughly validated and might cause system unavailability.
- It is recommended to perform the feature-related operations with the support from the TiDB team. If any misoperation is performed, it might be hard to recover the cluster.

Use this command to perform lossy recovery operations when permanently damaged replicas cause data to be unavailable. For example:

Execute Online Unsafe Recovery to remove permanently damaged stores:

```
>> unsafe remove-failed-stores 101,102,103
```

Success!

Show the current or historical state of Online Unsafe Recovery:

```
>> unsafe remove-failed-stores show
```

[

```
"Collecting cluster info from all alive stores, 10/12.",
"Stores that have reports to PD: 1, 2, 3, ...",
"Stores that have not reported to PD: 11, 12",
]
```

```
>> unsafe remove-failed-stores history
```

[

```
"Store reports collection:",
"Store 7: region 3 [start_key, end_key), {peer1, peer2, peer3} region 4
    ↪ ...",
"Store 8: region ...",
"...",
"Recovery Plan:",
"Store 7, creates: region 11, region 12, ...; updates: region 21, region
    ↪ 22, ... deletes: ... ",
"Store 8, ..."
"...",
"Execution Progress:",
"Store 10 finished,",
```

```
"Store 7 not yet finished",
"..." ,
]
```

11.7.2.5 Jq formatted JSON output usage

11.7.2.5.1 Simplify the output of store

```
>> store --jq=".stores[] .store | { id, address, state_name}"
{"id":1,"address":"127.0.0.1:20161","state_name":"Up"}
 {"id":30,"address":"127.0.0.1:20162","state_name":"Up"}
 ...
```

11.7.2.5.2 Query the remaining space of the node

```
>> store --jq=".stores[] | {id: .store.id, available: .status.available}"
 {"id":1,"available":"10 GiB"}
 {"id":30,"available":"10 GiB"}
 ...
```

11.7.2.5.3 Query all nodes whose status is not Up

```
>> store --jq='.stores[] .store | select(.state_name!="Up") | { id, address,
    ↪ state_name}'
```

```
{"id":1,"address":"127.0.0.1:20161""state_name":"Offline"}
 {"id":5,"address":"127.0.0.1:20162""state_name":"Offline"}
 ...
```

11.7.2.5.4 Query all TiFlash nodes

```
>> store --jq='.stores[] .store | select(.labels | length>0 and contains([{
    ↪ key": "engine", "value": "tiflash"}])) | { id, address, state_name}'
```

```
{"id":1,"address":"127.0.0.1:20161""state_name":"Up"}
 {"id":5,"address":"127.0.0.1:20162""state_name":"Up"}
 ...
```

11.7.2.5.5 Query the distribution status of the Region replicas

```
>> region --jq=".regions[] | {id: .id, peer_stores: [.peers[].store_id]}"
{"id":2,"peer_stores":[1,30,31]}
{"id":4,"peer_stores":[1,31,34]}
...
...
```

11.7.2.5.6 Filter Regions according to the number of replicas

For example, to filter out all Regions whose number of replicas is not 3:

```
>> region --jq=".regions[] | {id: .id, peer_stores: [.peers[].store_id] |
  → select(length != 3)}"
>{"id":12,"peer_stores":[30,32]}
>{"id":2,"peer_stores":[1,30,31,32]}
```

11.7.2.5.7 Filter Regions according to the store ID of replicas

For example, to filter out all Regions that have a replica on store30:

```
>> region --jq=".regions[] | {id: .id, peer_stores: [.peers[].store_id] |
  → select(any(.==30))}"
>{"id":6,"peer_stores":[1,30,31]}
>{"id":22,"peer_stores":[1,30,32]}
...
...
```

You can also find out all Regions that have a replica on store30 or store31 in the same way:

```
>> region --jq=".regions[] | {id: .id, peer_stores: [.peers[].store_id] |
  → select(any(.==(30,31)))}"
>{"id":16,"peer_stores":[1,30,34]}
>{"id":28,"peer_stores":[1,30,32]}
>{"id":12,"peer_stores":[30,32]}
...
...
```

11.7.2.5.8 Look for relevant Regions when restoring data

For example, when [store1, store30, store31] is unavailable at its downtime, you can find all Regions whose Down replicas are more than normal replicas:

```
>> region --jq=".regions[] | {id: .id, peer_stores: [.peers[].store_id] |
  → select(length as $total | map(if .==(1,30,31) then . else empty end)
  → | length>=$total-length) }"
>{"id":2,"peer_stores":[1,30,31,32]}
>{"id":12,"peer_stores":[30,32]}
```

```
{"id":14,"peer_stores":[1,30,32]}
...

```

Or when [store1, store30, store31] fails to start, you can find Regions where the data can be manually removed safely on store1. In this way, you can filter out all Regions that have a replica on store1 but don't have other DownPeers:

```
>> region --jq=".regions[] | {id: .id, peer_stores: [.peers[].store_id] |
  ↪ select(length>1 and any(.==1) and all(.!=(30,31)))}"
{"id":24,"peer_stores":[1,32,33]}
```

When [store30, store31] is down, find out all Regions that can be safely processed by creating the `remove-peer` Operator, that is, Regions with one and only DownPeer:

```
>> region --jq=".regions[] | {id: .id, remove_peer: [.peers[].store_id] |
  ↪ select(length>1) | map(if .==(30,31) then . else empty end) | select(
  ↪ length==1)}"
>{"id":12,"remove_peer":[30]}
>{"id":4,"remove_peer":[31]}
>{"id":22,"remove_peer":[30]}
...
```

11.7.3 TiDB Control User Guide

TiDB Control is a command-line tool of TiDB, usually used to obtain the status information of TiDB for debugging. This document introduces the features of TiDB Control and how to use these features.

11.7.3.1 Get TiDB Control

You can get TiDB Control by installing it using TiUP or by compiling it from source code.

Note:

It is recommended that the version of the Control tool you use is consistent with the version of the cluster.

11.7.3.1.1 Install TiDB Control using TiUP

After installing TiUP, you can use `tiup ctl tidb` command to get and execute TiDB Control.

11.7.3.1.2 Compile from source code

- Compilation environment requirement: [Go](#) Version 1.13 or later
- Compilation procedures: Go to the root directory of the [TiDB Control project](#), use the `make` command to compile, and generate `tidb-ctl`.
- Compilation documentation: you can find the help files in the `doc` directory; if the help files are lost or you want to update them, use the `make doc` command to generate the help files.

11.7.3.2 Usage introduction

This section describes how to use commands, subcommands, options, and flags in `tidb-ctl`.

- command: characters without `-` or `--`
- subcommand: characters without `-` or `--` that follow a command
- option: characters with `-` or `--`
- flag: characters exactly following a command/subcommand or option, passing value to the command/subcommand or option

Usage example: `tidb-ctl schema in mysql -n db`

- `schema`: the command
- `in`: the subcommand of `schema`
- `mysql`: the flag of `in`
- `-n`: the option
- `db`: the flag of `-n`

Currently, TiDB Control has the following subcommands:

- `tidb-ctl base64decode`: used for BASE64 decoding
- `tidb-ctl decoder`: used for KEY decoding
- `tidb-ctl etcd`: used for operating etcd
- `tidb-ctl log`: used to format the log file to expand the single-line stack information
- `tidb-ctl mvcc`: used to get the MVCC information
- `tidb-ctl region`: used to get the Region information
- `tidb-ctl schema`: used to get the schema information
- `tidb-ctl table`: used to get the table information

11.7.3.2.1 Get help

Use `tidb-ctl -h/--help` to get usage information.

TiDB Control consists of multiple layers of commands. You can use `-h/--help` after each command/subcommand to get its respective usage information.

The following example shows how to obtain the schema information:

Use `tidb-ctl schema -h` to get usage details. The `schema` command itself has two subcommands: `in` and `tid`.

- `in` is used to obtain the table schema of all tables in the database through the database name.
- `tid` is used to obtain the table schema by using the unique `table_id` in the whole database.

11.7.3.2.2 Global options

`tidb-ctl` has the following connection-related global options:

- `--host`: TiDB Service address (default 127.0.0.1)
- `--port`: TiDB status port (default 10080)
- `--pdhost`: PD Service address (default 127.0.0.1)
- `--pdport`: PD Service port (default 2379)
- `--ca`: The CA file path used for the TLS connection
- `--ssl-key`: The key file path used for the TLS connection
- `--ssl-cert`: The certificate file path used for the TLS connection

`--pdhost` and `--pdport` are mainly used in the `etcd` subcommand. For example, `tidb-ctl etcd ddlinfo`. If you do not specify the address and the port, the following default value is used:

- The default service address of TiDB and PD: 127.0.0.1. The service address must be an IP address.
- The default service port of TiDB: 10080.
- The default service port of PD: 2379.

11.7.3.2.3 The `schema` command

The `in` subcommand

`in` is used to obtain the table schema of all tables in the database through the database name.

```
tidb-ctl schema in <database name>
```

For example, running `tidb-ctl schema in mysql` returns the following result:

```
[
  {
    "id": 13,
    "name": {
      "O": "columns_priv",
      "L": "columns_priv"
    },
    ...
    "update_timestamp": 399494726837600268,
    "ShardRowIDBits": 0,
    "Partition": null
  }
]
```

The result is displayed in the JSON format. (The above output is truncated.)

- If you want to specify the table name, use `tidb-ctl schema in <database> -n <table name>` to filter.

For example, `tidb-ctl schema in mysql -n db` returns the table schema of the `db` table in the `mysql` database:

```
{
  "id": 9,
  "name": {
    "O": "db",
    "L": "db"
  },
  ...
  "Partition": null
}
```

(The above output is also truncated.)

If you do not want to use the default TiDB service address and port, use the `--host` and `--port` options to configure. For example, `tidb-ctl --host 172.16.55.88 --port 8898 schema in mysql -n db`.

The `tid` subcommand

`tid` is used to obtain the table schema by using the unique `table_id` in the whole database. You can use the `in` subcommand to get all table IDs of certain schema and use the `tid` subcommand to get the detailed table information.

For example, the table ID of `mysql.stat_meta` is 21. You can use `tidb-ctl schema tid -i 21` to obtain the detail of `mysql.stat_meta`.

```
{
  "id": 21,
  "name": {
    "0": "stats_meta",
    "L": "stats_meta"
  },
  "charset": "utf8mb4",
  "collate": "utf8mb4_bin",
  ...
}
```

Like the `in` subcommand, if you do not want to use the default TiDB service address and status port, use the `--host` and `--port` options to specify the host and port.

The `base64decode` command

`base64decode` is used to decode `base64` data.

```
tidb-ctl base64decode [base64_data]
tidb-ctl base64decode [db_name.table_name] [base64_data]
tidb-ctl base64decode [table_id] [base64_data]
```

1. Execute the following SQL statement to prepare the environment:

```
use test;
create table t (a int, b varchar(20),c datetime default
    ↪ current_timestamp , d timestamp default current_timestamp,
    ↪ unique index(a));
insert into t (a,b,c) values(1,"哈哈 hello",NULL);
alter table t add column e varchar(20);
```

2. Obtain MVCC data using the HTTP API interface:

```
$ curl "http://$IP:10080/mvcc/index/test/t/a/1?a=1"
{
  "info": {
    "writes": [
      {
        "start_ts": 407306449994645510,
        "commit_ts": 407306449994645513,
        "short_value": "AAAAAAAEE=" # The unique index a stores the
          ↪ handle id of the corresponding row.
      }
    ]
  }
}%
```

```
$ curl "http://$IP:10080/mvcc/key/test/t/1"
{
  "info": {
    "writes": [
      {
        "start_ts": 407306588892692486,
        "commit_ts": 407306588892692489,
        "short_value": "CAIIAggEAhj1k4j1k4ggaGVsbG8IBgAICAmAgIDwjYuu0Rk=" #
          ↪ Row data that handle id is 1.
      }
    ]
  }
}%
}
```

3. Decode handle id (uint64) using `base64decode`.

```
$ tidb-ctl base64decode AAAAAAAAEE=
hex: 0000000000000001
uint64: 1
```

4. Decode row data using base64decode.

```
$ ./tidb-ctl base64decode test.t
  ↪ CAIIAggEAhj1k4j1k4ggaGVsbG8IBgAICAmAgIDwjYuu0Rk=
a: 1
b: 哈哈 hello
c is NULL
d: 2019-03-28 05:35:30
e not found in data

# if the table id of test.t is 60, you can also use below command to do
  ↪ the same thing.
$ ./tidb-ctl base64decode 60
  ↪ CAIIAggEAhj1k4j1k4ggaGVsbG8IBgAICAmAgIDwjYuu0Rk=
a: 1
b: 哈哈 hello
c is NULL
d: 2019-03-28 05:35:30
e not found in data
```

11.7.3.2.4 The decoder command

- The following example shows how to decode the row key, similar to decoding the index key.

```
$ ./tidb-ctl decoder "t\x00\x00\x00\x00\x00\x00\x00\x00\x1c_r\x00\x00\x00\x00\x00\x00\x00\xfa"
format: table_row
table_id: -9223372036854775780 table_id: -9223372036854775780
row_id: -9223372036854775558 row_id: -9223372036854775558
```

- The following example shows how to decode value.

```
$ ./tidb-ctl decoder AhZoZWxsbyB3b3JsZAiAEA==
format: index_value
type: bigint, value: 1024  index_value[0]: {type: bytes, value: hello
                           ↪ world}
index_value[1]: {type: bigint, value: 1024}
```

11.7.3.2.5 The etcd command

- `tidb-ctl etcd ddlinfo` is used to obtain DDL information.
- `tidb-ctl etcd putkey KEY VALUE` is used to add KEY VALUE to etcd (All the KEYS are added to the `/tidb/ddl/all_schema_versions/` directory).

```
tidb-ctl etcd putkey "foo" "bar"
```

In fact, a key-value pair is added to the etcd whose KEY is `/tidb/ddl/all_schema_versions/foo` and VALUE is `bar`.

- `tidb-ctl etcd delkey` deletes the KEY in etcd. Only those KEYS with the `/tidb/ddl/fg/owner/` or `/tidb/ddl/all_schema_versions/` prefix can be deleted.

```
tidb-ctl etcd delkey "/tidb/ddl/fg/owner/foo"
tidb-ctl etcd delkey "/tidb/ddl/all_schema_versions/bar"
```

11.7.3.2.6 The log command

The stack information for the TiDB error log is in one line format. You could use `tidb-ctl log` to change its format to multiple lines.

11.7.3.2.7 The keyrange command

The `keyrange` subcommand is used to query the global or table-related key range information, which is output in the hexadecimal form.

- Execute the `tidb-ctl keyrange` command to check the global key range information:

```
tidb-ctl keyrange
```

```
global ranges:
  meta: (6d, 6e)
  table: (74, 75)
```

- Add the `--encode` option to display encoded keys (in the same format as in TiKV and PD):

```
tidb-ctl keyrange --encode
```

```
global ranges:
  meta: (6d00000000000000f8, 6e00000000000000f8)
  table: (7400000000000000f8, 7500000000000000f8)
```

- Execute the `tidb-ctl keyrange --database={db} --table={tbl}` command to check the global and table-related key range information:

```
tidb-ctl keyrange --database test --table ttt
```

```
global ranges:
  meta: (6d, 6e)
  table: (74, 75)
table ttt ranges: (NOTE: key range might be changed after DDL)
  table: (74800000000000002f, 748000000000000030)
  table indexes: (74800000000000002f5f69, 74800000000000002f5f72)
    index c2: (74800000000000002f5f69800000000000000001,
      ↢ 74800000000000002f5f69800000000000000002)
    index c3: (74800000000000002f5f69800000000000000002,
      ↢ 74800000000000002f5f69800000000000000003)
    index c4: (74800000000000002f5f69800000000000000003,
      ↢ 74800000000000002f5f69800000000000000004)
  table rows: (74800000000000002f5f72, 748000000000000030)
```

11.7.4 PD Recover User Guide

PD Recover is a disaster recovery tool of PD, used to recover the PD cluster which cannot start or provide services normally.

11.7.4.1 Compile from source code

- Go Version 1.13 or later is required because the Go modules are used.
- In the root directory of the [PD project](#), use the `make pd-recover` command to compile and generate `bin/pd-recover`.

Note:

Generally, you do not need to compile source code because the PD Control tool already exists in the released binary or Docker. However, developer users can refer to the instructions above for compiling source code.

11.7.4.2 Download TiDB installation package

To download the latest version of PD Recover, directly download the TiDB package, because PD Recover is included in the TiDB package.

Package name	OS	Architecture	SHA256
https://github.com/pingcap/tidb/releases	Linux	amd64	https://github.com/pingcap/tidb/releases
→ ://			→ ://
→ download			→ download
→ .			→ .
→ pingcap			→ pingcap
→ .			→ .
→ org			→ org
→ /			→ /
→ tidb			→ tidb
→ -{			→ -{
→ version			→ version
→ }-			→ }-
→ linux			→ linux
→ -			→ -
→ amd64			→ amd64
→ .			→ .
→ tar			→ sha256
→ .			→
→ gz			
→			
(pd-recover)			

Note:

{version} indicates the version number of TiDB. For example, if {version} is v5.3.0, the package download link is <https://download.pingcap.org/tidb-v5.3.0-linux-amd64.tar.gz>.

11.7.4.3 Quick Start

This section describes how to use PD Recover to recover a PD cluster.

11.7.4.3.1 Get cluster ID

The cluster ID can be obtained from the log of PD, TiKV or TiDB. To get the cluster ID, you can view the log directly on the server.

Get cluster ID from PD log (recommended)

To get the cluster ID from the PD log, run the following command:

```
cat {{/path/to}}/pd.log | grep "init cluster id"
```

```
[2019/10/14 10:35:38.880 +00:00] [INFO] [server.go:212] ["init cluster id"]
  ↪ [cluster-id=6747551640615446306]
...
```

Get cluster ID from TiDB log

To get the cluster ID from the TiDB log, run the following command:

```
cat {{/path/to}}/tidb.log | grep "init cluster id"
```

```
2019/10/14 19:23:04.688 client.go:161: [info] [pd] init cluster id
  ↪ 6747551640615446306
...
```

Get cluster ID from TiKV log

To get the cluster ID from the TiKV log, run the following command:

```
cat {{/path/to}}/tikv.log | grep "connect to PD cluster"
```

```
[2019/10/14 07:06:35.278 +00:00] [INFO] [tikv-server.rs:464] ["connect to PD
  ↪  cluster 6747551640615446306"]
...
```

11.7.4.3.2 Get allocated ID

The allocated ID value you specify must be larger than the currently largest allocated ID value. To get allocated ID, you can either get it from the monitor, or view the log directly on the server.

Get allocated ID from the monitor (recommended)

To get allocated ID from the monitor, you need to make sure that the metrics you are viewing are the metrics of **the last PD leader**, and you can get the largest allocated ID from the **Current ID allocation** panel in PD dashboard.

Get allocated ID from PD log

To get the allocated ID from the PD log, you need to make sure that the log you are viewing is the log of **the last PD leader**, and you can get the maximum allocated ID by running the following command:

```
cat {{/path/to}}/pd*.log | grep "idAllocator allocates a new id" | awk -F
    ↵ '=' '{print $2}' | awk -F']' '{print $1}' | sort -r -n | head -n 1
```

```
4000
```

```
...
```

Or you can simply run the above command in all PD servers to find the largest one.

11.7.4.3.3 Deploy a new PD cluster

Before deploying a new PD cluster, you need to stop the existing PD cluster and then delete the previous data directory which is specified by `--data-dir`.

11.7.4.3.4 Use pd-recover

```
./pd-recover -endpoints http://10.0.1.13:2379 -cluster-id
    ↵ 6747551640615446306 -alloc-id 10000
```

11.7.4.3.5 Restart the whole cluster

When you see the prompted information that the recovery is successful, restart the whole cluster.

11.7.4.4 FAQ

11.7.4.4.1 Multiple cluster IDs are found when getting the cluster ID

When a PD cluster is created, a new cluster ID is generated. You can determine the cluster ID of the old cluster by viewing the log.

11.7.4.4.2 The error dial tcp 10.0.1.13:2379: connect: connection refused is returned when executing pd-recover

The PD service is required when you execute `pd-recover`. Deploy and start the PD cluster before you use PD Recover.

11.8 Command Line Flags

11.8.1 Configuration Options

When you start the TiDB cluster, you can use command-line options or environment variables to configure it. This document introduces TiDB's command options. The default TiDB ports are 4000 for client requests and 10080 for status report.

11.8.1.1 --advertise-address

- The IP address through which to log into the TiDB server
- Default: “”
- This address must be accessible by the rest of the TiDB cluster and the user.

11.8.1.2 --config

- The configuration file
- Default: “”
- If you have specified the configuration file, TiDB reads the configuration file. If the corresponding configuration also exists in the command line options, TiDB uses the configuration in the command line options to overwrite that in the configuration file. For detailed configuration information, see [TiDB Configuration File Description](#).

11.8.1.3 --config-check

- Checks the validity of the configuration file and exits
- Default: false

11.8.1.4 --config-strict

- Enforces the validity of the configuration file
- Default: false

11.8.1.5 --cors

- Specifies the `Access-Control-Allow-Origin` value for Cross-Origin Request Sharing (CORS) request of the TiDB HTTP status service
- Default: “”

11.8.1.6 --host

- The host address that the TiDB server monitors
- Default: “0.0.0.0”
- The TiDB server monitors this address.
- The “0.0.0.0” address monitors all network cards by default. If you have multiple network cards, specify the network card that provides service, such as 192.168.100.113.

11.8.1.7 --enable-binlog

- Enables or disables TiDB binlog generation
- Default: false

11.8.1.8 -L

- The log level
- Default: “info”
- You can choose from “debug”, “info”, “warn”, “error”, or “fatal”.

11.8.1.9 --lease

- The duration of the schema lease. It is **dangerous** to change the value unless you know what you do.
- Default: 45s

11.8.1.10 --log-file

- The log file
- Default: “”
- If this option is not set, logs are output to “stderr”. If this option is set, logs are output to the corresponding file, which is automatically rotated in the early morning every day, and the previous file is renamed as a backup.

11.8.1.11 --log-slow-query

- The directory for the slow query log
- Default: “”
- If this option is not set, logs are output to the file specified by `--log-file` by default.

11.8.1.12 --metrics-addr

- The Prometheus Pushgateway address
- Default: “”
- Leaving it empty stops the Prometheus client from pushing.
- The format is `--metrics-addr=192.168.100.115:9091`.

11.8.1.13 --metrics-interval

- The Prometheus client push interval in seconds
- Default: 15s
- Setting the value to 0 stops the Prometheus client from pushing.

11.8.1.14 -P

- The monitoring port of TiDB services
- Default: “4000”
- The TiDB server accepts MySQL client requests from this port.

11.8.1.15 --path

- The path to the data directory for local storage engine like “unistore”
- For `--store = tikv`, you must specify the path; for `--store = unistore`, the default value is used if you do not specify the path.
- For the distributed storage engine like TiKV, `--path` specifies the actual PD address. Assuming that you deploy the PD server on 192.168.100.113:2379, 192.168.100.114:2379 and 192.168.100.115:2379, the value of `--path` is “192.168.100.113:2379, 192.168.100.114:2379, 192.168.100.115:2379”.
- Default: “/tmp/tidb”
- You can use `tidb-server --store=unistore --path=""` to enable a pure in-memory TiDB.

11.8.1.16 --proxy-protocol-networks

- The list of proxy server’s IP addresses allowed to connect to TiDB using the [PROXY protocol](#).
- Default: “”
- In general cases, when you access TiDB behind a reverse proxy, TiDB takes the IP address of the reverse proxy server as the IP address of the client. By enabling the PROXY protocol, reverse proxies that support this protocol such as HAProxy can pass the real client IP address to TiDB.

- After configuring this flag, TiDB allows the configured source IP address to connect to TiDB using the PROXY protocol; if a protocol other than PROXY is used, this connection will be denied. If this flag is left empty, no IP address can connect to TiDB using the PROXY protocol. The value can be the IP address (192.168.1.50) or CIDR (192.168.1.0/24) with , as the separator. * means any IP addresses.

Warning:

Use * with caution because it might introduce security risks by allowing a client of any IP address to report its IP address. In addition, using * might also cause the internal component that directly connects to TiDB (such as TiDB Dashboard) to be unavailable.

11.8.1.17 --proxy-protocol-header-timeout

- Timeout for the PROXY protocol header read
- Default: 5 (seconds)

Note:

Do not set the value to 0. Use the default value except for special situations.

11.8.1.18 --report-status

- Enables (`true`) or disables (`false`) the status report and pprof tool
- Default: `true`
- When set to `true`, this parameter enables metrics and pprof. When set to `false`, this parameter disables metrics and pprof.

11.8.1.19 --run-ddl

- To see whether the `tidb-server` runs DDL statements, and set when the number of `tidb-server` is over two in the cluster
- Default: `true`
- The value can be (`true`) or (`false`). (`true`) indicates the `tidb-server` runs DDL itself. (`false`) indicates the `tidb-server` does not run DDL itself.

11.8.1.20 --socket string

- The TiDB services use the unix socket file for external connections.
- Default: “”
- Use `/tmp/tidb.sock` to open the unix socket file.

11.8.1.21 --status

- The status report port for TiDB server
- Default: “10080”
- This port is used to get server internal data. The data includes [Prometheus metrics](#) and [pprof](#).
- Prometheus metrics can be accessed by `"http://host:status_port/metrics"`.
- pprof data can be accessed by `"http://host:status_port/debug/pprof"`.

11.8.1.22 --status-host

- The HOST used to monitor the status of TiDB service
- Default: `0.0.0.0`

11.8.1.23 --store

- Specifies the storage engine used by TiDB in the bottom layer
- Default: “unistore”
- You can choose “unistore” or “tikv”. (“unistore” is the local storage engine; “tikv” is a distributed storage engine)

11.8.1.24 --token-limit

- The number of sessions allowed to run concurrently in TiDB. It is used for traffic control.
- Default: 1000
- If the number of the concurrent sessions is larger than `token-limit`, the request is blocked and waiting for the operations which have been finished to release tokens.

11.8.1.25 -V

- Outputs the version of TiDB
- Default: “”

11.8.1.26 --plugin-dir

- The storage directory for plugins.
- Default: “/data/deploy/plugin”

11.8.1.27 --plugin-load

- The names of the plugins to be loaded, each separated by a comma.
- Default: “”

11.8.1.28 --affinity-cpus

- Sets the CPU affinity of TiDB servers, which is separated by commas. For example, “1,2,3”.
- Default: “”

11.8.1.29 --repair-mode

- Determines whether to enable the repair mode, which is only used in the data repair scenario.
- Default: false

11.8.1.30 --repair-list

- The names of the tables to be repaired in the repair mode.
- Default: “”

11.8.1.31 --require-secure-transport

- Determines whether to require the client to use the secure mode for data transport.
- Default: false

11.8.2 TiKV Configuration Flags

TiKV supports some readable unit conversions for command line parameters.

- File size (based on byte): KB, MB, GB, TB, PB (or lowercase)
- Time (based on ms): ms, s, m, h

11.8.2.1 **-A, --addr**

- The address that the TiKV server monitors
- Default: “127.0.0.1:20160”
- To deploy a cluster, you must use **--addr** to specify the IP address of the current host, such as “192.168.100.113:20160”. If the cluster is run on Docker, specify the IP address of Docker as “0.0.0.0:20160”.

11.8.2.2 **--advertise-addr**

- The server advertise address for client traffic from outside
- Default: \${addr}
- If the client cannot connect to TiKV through the **--addr** address because of Docker or NAT network, you must manually set the **--advertise-addr** address.
- For example, the internal IP address of Docker is 172.17.0.1, while the IP address of the host is 192.168.100.113 and the port mapping is set to **-p 20160:20160**. In this case, you can set **--advertise-addr** to “192.168.100.113:20160”. The client can find this service through 192.168.100.113:20160.

11.8.2.3 **--status-addr**

- The port through which the TiKV service status is listened
- Default: “20180”
- The Prometheus can access this status information via `http://host:status_port/` ↪ `metrics`.
- The Profile can access this status information via `http://host:status_port/debug` ↪ `/pprof/profile`.

11.8.2.4 **--advertise-status-addr**

- The address through which TiKV accesses service status from outside.
- Default: The value of **--status-addr** is used.
- If the client cannot connect to TiKV through the **--status-addr** address because of Docker or NAT network, you must manually set the **--advertise-status-addr** address.
- For example, the internal IP address of Docker is 172.17.0.1, while the IP address of the host is 192.168.100.113 and the port mapping is set to **-p 20180:20180**. In this case, set **--advertise-status-addr="192.168.100.113:20180"**. The client can find this service through 192.168.100.113:20180.

11.8.2.5 `--c, --config`

- The config file
- Default: “”
- If you set the configuration using the command line, the same setting in the config file will be overwritten.

11.8.2.6 `--capacity`

- The store capacity
- Default: 0 (unlimited)
- PD uses this flag to determine how to balance the TiKV servers. (Tip: you can use 10GB instead of 1073741824)

11.8.2.7 `--data-dir`

- The path to the data directory
- Default: “/tmp/tikv/store”

11.8.2.8 `-L`

- The log level
- Default: “info”
- You can choose from trace, debug, info, warn, error, or off.

11.8.2.9 `--log-file`

- The log file
- Default: “”
- If this flag is not set, logs will be written to stderr. Otherwise, logs will be stored in the log file which will be automatically rotated every day.

11.8.2.10 `--pd`

- The address list of PD servers
- Default: “”
- To make TiKV work, you must use the value of `--pd` to connect the TiKV server to the PD server. Separate multiple PD addresses using comma, for example “192.168.100.113:2379, 192.168.100.114:2379, 192.168.100.115:2379”.

11.8.3 TiFlash Command-Line Flags

This document introduces the command-line flags that you can use when you launch TiFlash.

11.8.3.1 `server --config-file`

- Specifies the path of the TiFlash configuration file
- Default: “”
- You must specify the configuration file. For detailed configuration items, refer to [TiFlash configuration parameters](#).

11.8.4 PD Configuration Flags

PD is configurable using command-line flags and environment variables.

11.8.4.1 `--advertise-client-urls`

- The list of advertise URLs for the client to access PD
- Default: "\${client-urls}"
- In some situations such as in the Docker or NAT network environment, if a client cannot access PD through the default client URLs listened to by PD, you must manually set the advertise client URLs.
- For example, the internal IP address of Docker is 172.17.0.1, while the IP address of the host is 192.168.100.113 and the port mapping is set to -p 2379:2379. In this case, you can set `--advertise-client-urls` to "http://192.168.100.113:2379". The client can find this service through "http://192.168.100.113:2379".

11.8.4.2 `--advertise-peer-urls`

- The list of advertise URLs for other PD nodes (peers) to access a PD node
- Default: "\${peer-urls}"
- In some situations such as in the Docker or NAT network environment, if the other nodes (peers) cannot access the PD node through the default peer URLs listened to by this PD node, you must manually set the advertise peer URLs.
- For example, the internal IP address of Docker is 172.17.0.1, while the IP address of the host is 192.168.100.113 and the port mapping is set to -p 2380:2380. In this case, you can set `--advertise-peer-urls` to "http://192.168.100.113:2380". The other PD nodes can find this service through "http://192.168.100.113:2380".

11.8.4.3 --client-urls

- The list of client URLs to be listened to by PD
- Default: "http://127.0.0.1:2379"
- When you deploy a cluster, you must specify the IP address of the current host as `--client-urls` (for example, "http://192.168.100.113:2379"). If the cluster runs on Docker, specify the IP address of Docker as "http://0.0.0.0:2379".

11.8.4.4 --peer-urls

- The list of peer URLs to be listened to by a PD node
- Default: "http://127.0.0.1:2380"
- When you deploy a cluster, you must specify `--peer-urls` as the IP address of the current host, such as "http://192.168.100.113:2380". If the cluster runs on Docker, specify the IP address of Docker as "http://0.0.0.0:2380".

11.8.4.5 --config

- The configuration file
- Default: “”
- If you set the configuration using the command line, the same setting in the configuration file will be overwritten.

11.8.4.6 --data-dir

- The path to the data directory
- Default: “default.\${name}”

11.8.4.7 --initial-cluster

- The initial cluster configuration for bootstrapping
- Default: "{name}=http://{advertise-peer-url}"
- For example, if name is "pd", and advertise-peer-urls is "http://192.168.100.113:2380" ↵ , the `initial-cluster` is "pd=http://192.168.100.113:2380".
- If you need to start three PD servers, the `initial-cluster` might be:

```
pd1=http://192.168.100.113:2380, pd2=http://192.168.100.114:2380, pd3  
↪ =192.168.100.115:2380
```

11.8.4.8 --join

- Join the cluster dynamically
- Default: “”
- If you want to join an existing cluster, you can use `--join="${advertise-client-urls}"`, the `advertise-client-url` is any existing PD’s, multiply advertise client urls are separated by comma.

11.8.4.9 -L

- The log level
- Default: “info”
- You can choose from debug, info, warn, error, or fatal.

11.8.4.10 --log-file

- The log file
- Default: “”
- If this flag is not set, logs will be written to stderr. Otherwise, logs will be stored in the log file which will be automatically rotated every day.

11.8.4.11 --log-rotate

- To enable or disable log rotation
- Default: true
- When the value is true, follow the [log.file] in PD configuration files.

11.8.4.12 --name

- The human-readable unique name for this PD member
- Default: “pd”
- If you want to start multiply PDs, you must use different name for each one.

11.8.4.13 --cacert

- The file path of CA, used to enable TLS
- Default: “”

11.8.4.14 --cert

- The path of the PEM file including the X509 certificate, used to enable TLS
- Default: “”

11.8.4.15 --key

- The path of the PEM file including the X509 key, used to enable TLS
- Default: “”

11.8.4.16 --metrics-addr

- The address of Prometheus Pushgateway, which does not push data to Prometheus by default.
- Default: “”

11.9 Configuration File Parameters

11.9.1 TiDB Configuration File

The TiDB configuration file supports more options than command-line parameters. You can download the default configuration file `config.toml.example` and rename it to `config → .toml`. This document describes only the options that are not involved in `command line options`.

11.9.1.0.1 split-table

- Determines whether to create a separate Region for each table.
- Default value: `true`
- It is recommended to set it to `false` if you need to create a large number of tables.

11.9.1.0.2 token-limit

- The number of sessions that can execute requests concurrently.
- Type: Integer
- Default value: 1000
- Minimum value: 1
- Maximum Value (64-bit platforms): 18446744073709551615
- Maximum Value (32-bit platforms): 4294967295

11.9.1.0.3 mem-quota-query

- The maximum memory available for a single SQL statement.
- Default value: 1073741824 (in bytes)
- Note: When you upgrade the cluster from v2.0.x or v3.0.x to v4.0.9 or later versions, the default value of this configuration is 34359738368.
- Requests that require more memory than this value are handled based on the behavior defined by `oom-action`.
- This value is the initial value of the system variable `tidb_mem_quota_query`.

11.9.1.0.4 oom-use-tmp-storage

- Controls whether to enable the temporary storage for some operators when a single SQL statement exceeds the memory quota specified by `mem-quota-query`.
- Default value: `true`

11.9.1.0.5 tmp-storage-path

- Specifies the temporary storage path for some operators when a single SQL statement exceeds the memory quota specified by `mem-quota-query`.
- Default value: <temporary directory of OS>/<OS user ID>_tidb/MC4wLjAuMD0OMDAwLzAuMC4wL
↪ =/tmp-storage. MC4wLjAuMD0OMDAwLzAuMC4wLjA6MTAw0DA= is the Base64 encoding result of <host>:<port>/<statusHost>:<statusPort>.
- This configuration takes effect only when `oom-use-tmp-storage` is `true`.

11.9.1.0.6 tmp-storage-quota

- Specifies the quota for the storage in `tmp-storage-path`. The unit is byte.
- When a single SQL statement uses a temporary disk and the total volume of the temporary disk of the TiDB server exceeds this configuration value, the current SQL operation is cancelled and the `Out of Global Storage Quota!` error is returned.
- When the value of this configuration is smaller than 0, the above check and limit do not apply.
- Default value: -1
- When the remaining available storage in `tmp-storage-path` is lower than the value defined by `tmp-storage-quota`, the TiDB server reports an error when it is started, and exits.

11.9.1.0.7 oom-action

- Specifies what operation TiDB performs when a single SQL statement exceeds the memory quota specified by `mem-quota-query` and cannot be spilled over to disk.
- Default value: "cancel" (In TiDB v4.0.2 and earlier versions, the default value is "log")
- The valid options are "log" and "cancel". When `oom-action="log"`, it prints the log only. When `oom-action="cancel"`, it cancels the operation and outputs the log.

11.9.1.0.8 lower-case-table-names

- Configures the value of the `lower-case-table-names` system variable.
- Default value: 2

- For details, see the [MySQL description](#) of this variable.

Note:

Currently, TiDB only supports setting the value of this option to 2. This means it is case-sensitive when you save a table name, but case-insensitive when you compare table names. The comparison is based on the lower case.

11.9.1.0.9 `lease`

- The timeout of the DDL lease.
- Default value: `45s`
- Unit: second

11.9.1.0.10 `compatible-kill-query`

- Determines whether to set the `KILL` statement to be MySQL compatible.
- Default value: `false`
- The behavior of `KILL xxx` in TiDB differs from the behavior in MySQL. TiDB requires the `TIDB` keyword, namely, `KILL TIDB xxx`. If `compatible-kill-query` is set to `true`, the `TIDB` keyword is not needed.
- This distinction is important because the default behavior of the MySQL command-line client, when the user hits `Ctrl+C`, is to create a new connection to the backend and execute the `KILL` statement in that new connection. If a load balancer or proxy has sent the new connection to a different TiDB server instance than the original session, the wrong session could be terminated, which could cause interruption to applications using the cluster. Enable `compatible-kill-query` only if you are certain that the connection you refer to in your `KILL` statement is on the same server to which you send the `KILL` statement.

11.9.1.0.11 `check-mb4-value-in-utf8`

- Determines whether to enable the `utf8mb4` character check. When this feature is enabled, if the character set is `utf8` and the `mb4` characters are inserted in `utf8`, an error is returned.
- Default value: `false`

11.9.1.0.12 `treat-old-version-utf8-as-utf8mb4`

- Determines whether to treat the `utf8` character set in old tables as `utf8mb4`.
- Default value: `true`

11.9.1.0.13 alter-primary-key (Deprecated)

- Determines whether to add or remove the primary key constraint to or from a column.
- Default value: `false`
- With this default setting, adding or removing the primary key constraint is not supported. You can enable this feature by setting `alter-primary-key` to `true`. However, if a table already exists before the switch is on, and the data type of its primary key column is an integer, dropping the primary key from the column is not possible even if you set this configuration item to `true`.

Note:

This configuration item has been deprecated, and currently takes effect only when the value of `@tidb_enable_clustered_index` is `INT_ONLY`. If you need to add or remove the primary key, use the `NONCLUSTERED` keyword instead when creating the table. For more details about the primary key of the `CLUSTERED` type, refer to [clustered index](#).

11.9.1.0.14 server-version

- Modifies the version string returned by TiDB in the following situations:
 - When the built-in `VERSION()` function is used.
 - When TiDB establishes the initial connection to the client and returns the initial handshake packet with version string of the server. For details, see [MySQL Initial Handshake Packet](#).
- Default value: “”
- By default, the format of the TiDB version string is `5.7.${mysql_latest_minor_version} → }-TiDB-${tidb_version}`.

11.9.1.0.15 repair-mode

- Determines whether to enable the untrusted repair mode. When the `repair-mode` is set to `true`, bad tables in the `repair-table-list` cannot be loaded.
- Default value: `false`
- The `repair` syntax is not supported by default. This means that all tables are loaded when TiDB is started.

11.9.1.0.16 repair-table-list

- `repair-table-list` is only valid when `repair-mode` is set to `true`. `repair-table-list` is a list of bad tables that need to be repaired in an instance. An example of the list is: `["db.table1", "db.table2"]`.
- Default value: `[]`
- The list is empty by default. This means that there are no bad tables that need to be repaired.

11.9.1.0.17 new_collations_enabled_on_first_bootstrap

- Enables or disables the new collation support.
- Default value: `false`
- Note: This configuration takes effect only for the TiDB cluster that is first initialized. After the initialization, you cannot use this configuration item to enable or disable the new collation support. When a TiDB cluster is upgraded to v4.0, because the cluster has been initialized before, both `true` and `false` values of this configuration item are taken as `false`.

11.9.1.0.18 max-server-connections

- The maximum number of concurrent client connections allowed in TiDB. It is used to control resources.
- Default value: 0
- By default, TiDB does not set limit on the number of concurrent client connections. When the value of this configuration item is greater than 0 and the number of actual client connections reaches this value, the TiDB server rejects new client connections.

11.9.1.0.19 max-index-length

- Sets the maximum allowable length of the newly created index.
- Default value: 3072
- Unit: byte
- Currently, the valid value range is [3072, 3072*4]. MySQL and TiDB (version < v3.0.11) do not have this configuration item, but both limit the length of the newly created index. This limit in MySQL is 3072. In TiDB (version =< 3.0.7), this limit is 3072*4. In TiDB (3.0.7 < version < 3.0.11), this limit is 3072. This configuration is added to be compatible with MySQL and earlier versions of TiDB.

11.9.1.0.20 table-column-count-limit New in v5.0

- Sets the limit on the number of columns in a single table.
- Default value: 1017
- Currently, the valid value range is [1017, 4096].

11.9.1.0.21 `index-limit` New in v5.0

- Sets the limit on the number of indexes in a single table.
- Default value: `64`
- Currently, the valid value range is `[64, 512]`.

11.9.1.0.22 `enable-telemetry` New in v4.0.2

- Enables or disables the telemetry collection in TiDB.
- Default value: `true`
- When this configuration is set to `false` on all TiDB instances, the telemetry collection in TiDB is disabled and the `tidb_enable_telemetry` system variable does not take effect. See [Telemetry](#) for details.

11.9.1.0.23 `enable-tcp4-only` New in v5.0

- Enables or disables listening on TCP4 only.
- Default value: `false`
- Enabling this option is useful when TiDB is used with LVS for load balancing because the [real client IP from the TCP header](#) can be correctly parsed by the “tcp4” protocol.

11.9.1.0.24 `enable-enum-length-limit` New in v5.0

- Determines whether to limit the maximum length of a single ENUM element and a single SET element.
- Default value: `true`
- When this configuration value is `true`, the maximum length of a single ENUM element and a single SET element is 255 characters, which is compatible with [MySQL 8.0](#). When this configuration value is `false`, there is no limit on the length of a single element, which is compatible with TiDB (earlier than v5.0).

`graceful-wait-before-shutdown` New in v5.0

- Specifies the number of seconds that TiDB waits when you shut down the server, which allows the clients to disconnect.
- Default value: 0
- When TiDB is waiting for shutdown (in the grace period), the HTTP status will indicate a failure, which allows the load balancers to reroute traffic.

11.9.1.1 Log

Configuration items related to log.

11.9.1.1.1 level

- Specifies the log output level.
- Value options: `debug`, `info`, `warn`, `error`, and `fatal`.
- Default value: `info`

11.9.1.1.2 format

- Specifies the log output format.
- Value options: `json` and `text`.
- Default value: `text`

11.9.1.1.3 enable-timestamp

- Determines whether to enable timestamp output in the log.
- Default value: `true`
- If you set the value to `false`, the log does not output timestamp.

Note:

To be backward compatible, the initial `disable-timestamp` configuration item remains valid. But if the value of `disable-timestamp` semantically conflicts with the value of `enable-timestamp` (for example, if both `enable-timestamp` and `disable-timestamp` are set to `true`), TiDB ignores the value for `disable-timestamp`. In later versions, the `disable-timestamp` configuration will be removed.

Discard `disable-timestamp` and use `enable-timestamp` which is semantically easier to understand.

11.9.1.1.4 enable-slow-log

- Determines whether to enable the slow query log.
- Default value: `true`
- To enable the slow query log, set `enable-slow-log` to `true`. Otherwise, set it to `false`.

11.9.1.1.5 slow-query-file

- The file name of the slow query log.
- Default value: `tidb-slow.log`
- The format of the slow log is updated in TiDB v2.1.8, so the slow log is output to the slow log file separately. In versions before v2.1.8, this variable is set to “” by default.
- After you set it, the slow query log is output to this file separately.

11.9.1.1.6 slow-threshold

- Outputs the threshold value of consumed time in the slow log.
- Default value: 300ms
- If the value in a query is larger than the default value, it is a slow query and is output to the slow log.

11.9.1.1.7 record-plan-in-slow-log

- Determines whether to record execution plans in the slow log.
- Default value: 1
- 0 means to disable, and 1 (by default) means to enable. The value of this parameter is the initial value of the `tidb_record_plan_in_slow_log` system variable.

11.9.1.1.8 expensive-threshold

- Outputs the threshold value of the number of rows for the `expensive` operation.
- Default value: 10000
- When the number of query rows (including the intermediate results based on statistics) is larger than this value, it is an `expensive` operation and outputs log with the [→ EXPENSIVE_QUERY] prefix.

11.9.1.1.9 query-log-max-len

- The maximum length of SQL output.
- Default value: 4096
- When the length of the statement is longer than `query-log-max-len`, the statement is truncated to output.

11.9.1.2 log.file

Configuration items related to log files.

filename

- The file name of the general log file.
- Default value: “”
- If you set it, the log is output to this file.

max-size

- The size limit of the log file.
- Default value: 300
- Unit: MB
- The maximum value is 4096.

max-days

- The maximum number of days that the log is retained.
- Default value: 0
- The log is retained by default. If you set the value, the expired log is cleaned up after max-days.

max-backups

- The maximum number of retained logs.
- Default value: 0
- All the log files are retained by default. If you set it to 7, seven log files are retained at maximum.

11.9.1.3 Security

Configuration items related to security.

11.9.1.3.1 require-secure-transport

- Determines whether to require the client to use the secure mode for data transport.
- Default value: `false`

11.9.1.3.2 enable-sem

- Enables the Security Enhanced Mode (SEM).
- Default value: `false`
- The status of SEM is available via the system variable `tidb_enable_enhanced_security` ↗ .

11.9.1.3.3 `ssl-ca`

- The file path of the trusted CA certificate in the PEM format.
- Default value: “”
- If you set this option and `--ssl-cert`, `--ssl-key` at the same time, TiDB authenticates the client certificate based on the list of trusted CAs specified by this option when the client presents the certificate. If the authentication fails, the connection is terminated.
- If you set this option but the client does not present the certificate, the secure connection continues without client certificate authentication.

11.9.1.3.4 `ssl-cert`

- The file path of the SSL certificate in the PEM format.
- Default value: “”
- If you set this option and `--ssl-key` at the same time, TiDB allows (but not forces) the client to securely connect to TiDB using TLS.
- If the specified certificate or private key is invalid, TiDB starts as usual but cannot receive secure connection.

11.9.1.3.5 `ssl-key`

- The file path of the SSL certificate key in the PEM format, that is, the private key of the certificate specified by `--ssl-cert`.
- Default value: “”
- Currently, TiDB does not support loading the private keys protected by passwords.

11.9.1.3.6 `cluster-ssl-ca`

- The CA root certificate used to connect TiKV or PD with TLS.
- Default value: “”

11.9.1.3.7 `cluster-ssl-cert`

- The path of the SSL certificate file used to connect TiKV or PD with TLS.
- Default value: “”

11.9.1.3.8 `cluster-ssl-key`

- The path of the SSL private key file used to connect TiKV or PD with TLS.
- Default value: “”

11.9.1.3.9 spilled-file-encryption-method

- Determines the encryption method used for saving the spilled files to disk.
- Default value: "plaintext", which disables encryption.
- Optional values: "plaintext" and "aes128-ctr"

11.9.1.3.10 auto-tls

- Determines whether to automatically generate the TLS certificates on startup.
- Default value: `false`

11.9.1.4 Performance

Configuration items related to performance.

11.9.1.4.1 max-procs

- The number of CPUs used by TiDB.
- Default value: 0
- The default 0 indicates using all the CPUs on the machine. You can also set it to n, and then TiDB uses n CPUs.

11.9.1.4.2 server-memory-quota New in v4.0.9

Warning:

`server-memory-quota` is still an experimental feature. It is **NOT** recommended that you use it in a production environment.

- The memory usage limit of tidb-server instances. This configuration item completely supersedes the previous `max-memory`.
- Default value: 0 (in bytes), which means no memory limit.

11.9.1.4.3 memory-usage-alarm-ratio New in v4.0.9

- TiDB triggers an alarm when the memory usage of tidb-server instance exceeds a certain threshold. The valid value for this configuration item ranges from 0 to 1. If it is configured as 0 or 1, this alarm feature is disabled.
- Default value: 0.8

- When the memory usage alarm is enabled, if `server-memory-quota` is not set, then the threshold of memory usage is the `'memory-usage-alarm-ratio'` value * the \rightarrow system memory size; if `server-memory-quota` is set to a value greater than 0, then the threshold of memory usage is the `'memory-usage-alarm-ratio'` value * \rightarrow the `'server-memory-quota'` value.
- When TiDB detects that the memory usage of the tidb-server instance exceeds the threshold, it considers that there might be a risk of OOM. Therefore, it records ten SQL statements with the highest memory usage, ten SQL statements with the longest running time, and the heap profile among all SQL statements currently being executed to the directory `tmp-storage-path/record` and outputs a log containing the keyword `tidb-server has the risk of OOM`.
- The value of this configuration item is the initial value of the system variable `tidb_memory_usage_alarm_ratio`.

11.9.1.4.4 max-txn-ttl

- The longest time that a single transaction can hold locks. If this time is exceeded, the locks of a transaction might be cleared by other transactions so that this transaction cannot be successfully committed.
- Default value: 3600000
- Unit: Millisecond
- The transaction that holds locks longer than this time can only be committed or rolled back. The commit might not be successful.

11.9.1.4.5 committer-concurrency

- The number of goroutines for requests related to executing commit in the commit phase of the single transaction.
- Default value: 128
- If the transaction to commit is too large, the waiting time for the flow control queue when the transaction is committed might be too long. In this situation, you can increase the configuration value to speed up the commit.

11.9.1.4.6 stmt-count-limit

- The maximum number of statements allowed in a single TiDB transaction.
- Default value: 5000
- If a transaction does not roll back or commit after the number of statements exceeds `stmt-count-limit`, TiDB returns the `statement count 5001 exceeds the → transaction limitation, autocommit = false` error. This configuration takes effect **only** in the retryable optimistic transaction. If you use the pessimistic transaction or have disabled the transaction retry, the number of statements in a transaction is not limited by this configuration.

11.9.1.4.7 `txn-entry-size-limit` New in v5.0

- The size limit of a single row of data in TiDB.
- Default value: 6291456 (in bytes)
- The size limit of a single key-value record in a transaction. If the size limit is exceeded, TiDB returns the `entry too large` error. The maximum value of this configuration item does not exceed 125829120 (120 MB).
- Note that TiKV has a similar limit. If the data size of a single write request exceeds `raft-entry-max-size`, which is 8 MB by default, TiKV refuses to process this request. When a table has a row of large size, you need to modify both configurations at the same time.

11.9.1.4.8 `txn-total-size-limit`

- The size limit of a single transaction in TiDB.
- Default value: 104857600 (in bytes)
- In a single transaction, the total size of key-value records cannot exceed this value. The maximum value of this parameter is 10737418240 (10 GB). Note that if you have used the binlog to serve the downstream consumer Kafka (such as the `arbiter` cluster), the value of this parameter must be no more than 1073741824 (1 GB). This is because 1 GB is the upper limit of a single message size that Kafka can process. Otherwise, an error is returned if this limit is exceeded.

11.9.1.4.9 `tcp-keep-alive`

- Determines whether to enable `keepalive` in the TCP layer.
- Default value: `true`

11.9.1.4.10 `tcp-no-delay`

- Determines whether to enable `TCP_NODELAY` at the TCP layer. After it is enabled, TiDB disables the Nagle algorithm in the TCP/IP protocol and allows sending small data packets to reduce network latency. This is suitable for latency-sensitive applications with a small transmission volume of data.
- Default value: `true`

11.9.1.4.11 `cross-join`

- Default value: `true`
- TiDB supports executing the `JOIN` statement without any condition (the `WHERE` field) of both sides tables by default; if you set the value to `false`, the server refuses to execute when such a `JOIN` statement appears.

11.9.1.4.12 stats-lease

- The time interval of reloading statistics, updating the number of table rows, checking whether it is needed to perform the automatic analysis, using feedback to update statistics and loading statistics of columns.
- Default value: 3s
 - At intervals of `stats-lease` time, TiDB checks the statistics for updates and updates them to the memory if updates exist.
 - At intervals of $20 * \text{stats-lease}$ time, TiDB updates the total number of rows generated by DML and the number of modified rows to the system table.
 - At intervals of `stats-lease`, TiDB checks for tables and indexes that need to be automatically analyzed.
 - At intervals of `stats-lease`, TiDB checks for column statistics that need to be loaded to the memory.
 - At intervals of $200 * \text{stats-lease}$, TiDB writes the feedback cached in the memory to the system table.
 - At intervals of $5 * \text{stats-lease}$, TiDB reads the feedback in the system table, and updates the statistics cached in the memory.
- When `stats-lease` is set to 0s, TiDB periodically reads the feedback in the system table, and updates the statistics cached in the memory every three seconds. But TiDB no longer automatically modifies the following statistics-related system tables:
 - `mysql.stats_meta`: TiDB no longer automatically records the number of table rows that are modified by the transaction and updates it to this system table.
 - `mysql.stats_histograms/mysql.stats_buckets` and `mysql.stats_top_n`: TiDB no longer automatically analyzes and proactively updates statistics.
 - `mysql.stats_feedback`: TiDB no longer updates the statistics of the tables and indexes according to a part of statistics returned by the queried data.

11.9.1.4.13 run-auto-analyze

- Determines whether TiDB executes automatic analysis.
- Default value: `true`

11.9.1.4.14 feedback-probability

- The probability that TiDB collects the feedback statistics of each query.
- Default value: 0
- This feature is disabled by default, and it is not recommended to enable this feature. If it is enabled, TiDB collects the feedback of each query at the probability of `feedback -probability`, to update statistics.

11.9.1.4.15 `query-feedback-limit`

- The maximum pieces of query feedback that can be cached in memory. Extra pieces of feedback that exceed this limit are discarded.
- Default value: 1024

11.9.1.4.16 `pseudo-estimate-ratio`

- The ratio of (number of modified rows)/(total number of rows) in a table. If the value is exceeded, the system assumes that the statistics have expired and the pseudo statistics will be used.
- Default value: 0.8
- The minimum value is 0 and the maximum value is 1.

11.9.1.4.17 `force-priority`

- Sets the priority for all statements.
- Default: NO_PRIORITY
- Optional values: NO_PRIORITY, LOW_PRIORITY, HIGH_PRIORITY and DELAYED.

11.9.1.4.18 `distinct-agg-push-down`

- Determines whether the optimizer executes the operation that pushes down the aggregation function with Distinct (such as `select count(distinct a)from t`) to Coprocessors.
- Default: false
- This variable is the initial value of the system variable `tidb_opt_distinct_agg_push_down` ↗ .

11.9.1.4.19 `nested-loop-join-cache-capacity`

- The maximum memory usage for the Least Recently Used (LRU) algorithm of the nested loop join cache (in bytes).
- Default value: 20971520
- When `nested-loop-join-cache-capacity` is set to 0, nested loop join cache is disabled by default. When the LRU size is larger than the value of `nested-loop-join-cache-capacity`, the elements in the LRU are removed.

11.9.1.4.20 `enforce-mpp`

- Determines whether to ignore the optimizer's cost estimation and to forcibly use TiFlash's MPP mode for query execution.
- Default value: `false`
- This configuration item controls the initial value of `tidb_enforce_mpp`. For example, when this configuration item is set to `true`, the default value of `tidb_enforce_mpp` is ON.

11.9.1.5 `prepared-plan-cache`

The Plan Cache configuration of the PREPARE statement.

11.9.1.5.1 `enabled`

- Determines whether to enable Plan Cache of the PREPARE statement.
- Default value: `false`

11.9.1.5.2 `capacity`

- The number of cached statements.
- Default value: 1000
- The type is UINT. Values less than 0 are converted to large integers.

11.9.1.5.3 `memory-guard-ratio`

- It is used to prevent `performance.max-memory` from being exceeded. When $\hookrightarrow -memory * (1 - prepared-plan-cache.memory-guard-ratio)$ is exceeded, the elements in the LRU are removed.
- Default value: 0.1
- The minimum value is 0; the maximum value is 1.

11.9.1.6 `tikv-client`

11.9.1.6.1 `grpc-connection-count`

- The maximum number of connections established with each TiKV.
- Default value: 4

11.9.1.6.2 `grpc-keepalive-time`

- The `keepalive` time interval of the RPC connection between TiDB and TiKV nodes. If there is no network packet within the specified time interval, the gRPC client executes `ping` command to TiKV to see if it is alive.
- Default: 10
- Unit: second

11.9.1.6.3 `grpc-keepalive-timeout`

- The timeout of the RPC `keepalive` check between TiDB and TiKV nodes.
- Default value: 3
- Unit: second

11.9.1.6.4 `commit-timeout`

- The maximum timeout when executing a transaction commit.
- Default value: 41s
- It is required to set this value larger than twice of the Raft election timeout.

11.9.1.6.5 `max-batch-size`

- The maximum number of RPC packets sent in batch. If the value is not 0, the `BatchCommands` API is used to send requests to TiKV, and the RPC latency can be reduced in the case of high concurrency. It is recommended that you do not modify this value.
- Default value: 128

11.9.1.6.6 `max-batch-wait-time`

- Waits for `max-batch-wait-time` to encapsulate the data packets into a large packet in batch and send it to the TiKV node. It is valid only when the value of `tikv-client.max-batch-size` is greater than 0. It is recommended not to modify this value.
- Default value: 0
- Unit: nanoseconds

11.9.1.6.7 `batch-wait-size`

- The maximum number of packets sent to TiKV in batch. It is recommended not to modify this value.
- Default value: 8
- If the value is 0, this feature is disabled.

11.9.1.6.8 `overload-threshold`

- The threshold of the TiKV load. If the TiKV load exceeds this threshold, more batch packets are collected to relieve the pressure of TiKV. It is valid only when the value of `tikv-client.max-batch-size` is greater than 0. It is recommended not to modify this value.
- Default value: 200

11.9.1.7 `tikv-client.copr-cache` New in v4.0.0

This section introduces configuration items related to the Coprocessor Cache feature.

11.9.1.7.1 `capacity-mb`

- The total size of the cached data. When the cache space is full, old cache entries are evicted. When the value is 0.0, the Coprocessor Cache feature is disabled.
- Default value: 1000.0
- Unit: MB
- Type: Float

11.9.1.8 `txn-local-latches`

Configuration related to the transaction latch. It is recommended to enable it when many local transaction conflicts occur.

11.9.1.8.1 `enabled`

- Determines whether to enable the memory lock of transactions.
- Default value: `false`

11.9.1.8.2 `capacity`

- The number of slots corresponding to Hash, which automatically adjusts upward to an exponential multiple of 2. Each slot occupies 32 Bytes of memory. If set too small, it might result in slower running speed and poor performance in the scenario where data writing covers a relatively large range (such as importing data).
- Default value: 2048000

11.9.1.9 `binlog`

Configurations related to TiDB Binlog.

11.9.1.9.1 enable

- Enables or disables binlog.
- Default value: `false`

11.9.1.9.2 write-timeout

- The timeout of writing binlog into Pump. It is not recommended to modify this value.
- Default: `15s`
- unit: second

11.9.1.9.3 ignore-error

- Determines whether to ignore errors occurred in the process of writing binlog into Pump. It is not recommended to modify this value.
- Default value: `false`
- When the value is set to `true` and an error occurs, TiDB stops writing binlog and add 1 to the count of the `tidb_server_critical_error_total` monitoring item. When the value is set to `false`, the binlog writing fails and the entire TiDB service is stopped.

11.9.1.9.4 binlog-socket

- The network address to which binlog is exported.
- Default value: “”

11.9.1.9.5 strategy

- The strategy of Pump selection when binlog is exported. Currently, only the `hash` and `range` methods are supported.
- Default value: `range`

11.9.1.10 status

Configuration related to the status of TiDB service.

11.9.1.10.1 report-status

- Enables or disables the HTTP API service.
- Default value: `true`

11.9.1.10.2 record-db-qps

- Determines whether to transmit the database-related QPS metrics to Prometheus.
- Default value: `false`

11.9.1.11 stmt-summary New in v3.0.4

Configurations related to [statement summary tables](#).

11.9.1.11.1 max-stmt-count

- The maximum number of SQL categories allowed to be saved in [statement summary tables](#).
- Default value: 3000

11.9.1.11.2 max-sql-length

- The longest display length for the `DIGEST_TEXT` and `QUERY_SAMPLE_TEXT` columns in [statement summary tables](#).
- Default value: 4096

11.9.1.12 pessimistic-txn

For pessimistic transaction usage, refer to [TiDB Pessimistic Transaction Mode](#).

11.9.1.12.1 max-retry-count

- The maximum number of retries of each statement in pessimistic transactions. If the number of retries exceeds this limit, an error occurs.
- Default value: 256

11.9.1.12.2 deadlock-history-capacity

- The maximum number of deadlock events that can be recorded in the [INFORMATION_SCHEMA](#) \hookrightarrow `.DEADLOCKS` table of a single TiDB server. If this table is in full volume and an additional deadlock event occurs, the earliest record in the table will be removed to make place for the newest error.
- Default value: 10
- Minimum value: 0
- Maximum value: 10000

11.9.1.12.3 deadlock-history-collect-retryable

- Controls whether the `INFORMATION_SCHEMA.DEADLOCKS` table collects the information of retryable deadlock errors. For the description of retryable deadlock errors, see [Retryable deadlock errors](#).
- Default value: `false`

11.9.1.13 experimental

The `experimental` section, introduced in v3.1.0, describes the configurations related to the experimental features of TiDB.

11.9.1.13.1 allow-expression-index New in v4.0.0

- Controls whether an expression index can be created. Since TiDB v5.2.0, if the function in an expression is safe, you can create an expression index directly based on this function without enabling this configuration. If you want to create an expression index based on other functions, you can enable this configuration, but correctness issues might exist. By querying the `tidb_allow_function_for_expression_index` variable, you can get the functions that are safe to be directly used for creating an expression.
- Default value: `false`

11.9.2 TiKV Configuration File

The TiKV configuration file supports more options than command-line parameters. You can find the default configuration file in `etc/config-template.toml` and rename it to `config → .toml`.

This document only describes the parameters that are not included in command-line parameters. For more details, see [command-line parameter](#).

11.9.2.1 Global configuration

11.9.2.1.1 abort-on-panic

- Sets whether to call `abort()` to exit the process when TiKV panics. This option affects whether TiKV allows the system to generate core dump files.
 - If the value of this configuration item is `false`, when TiKV panics, it calls `exit()` to exit the process.

- If the value of this configuration item is `true`, when TiKV panics, TiKV calls `abort()` to exit the process. At this time, TiKV allows the system to generate core dump files when exiting. To generate the core dump file, you also need to perform the system configuration related to core dump (for example, setting the size limit of the core dump file via `ulimit -c` command, and configure the core dump path. Different operating systems have different related configurations). To avoid the core dump files occupying too much disk space and causing insufficient TiKV disk space, it is recommended to set the core dump generation path to a disk partition different to that of TiKV data.

- Default value: `false`

11.9.2.1.2 `log-level`

- The log level
- Value options: “trace”, “debug”, “info”, “warning”, “error”, “critical”
- Default value: “info”

11.9.2.1.3 `log-file`

- The log file. If this configuration is not set, logs are output to “stderr” by default.
- Default value: “”

11.9.2.1.4 `log-format`

- The log format
- Value options: “json”, “text”
- Default value: “text”

11.9.2.1.5 `log-rotation-timespan`

- The timespan between log rotations. When this timespan passes, log files are rotated, that is, a timestamp is appended to the file name of the current log file, and a new file is created.
- Default value: “24h”

11.9.2.1.6 `log-rotation-size`

- The size of a log file that triggers log rotation. Once the size of a log file is bigger than the specified threshold value, log files are rotated. The old log file is placed into the new file, and the new file name is the old file name with a timestamp suffix.
- Default value: “300MB”

11.9.2.1.7 slow-log-file

- The file to store slow logs
- If this configuration is not set but `log-file` is set, slow logs are output to the log file specified by `log-file`. If neither `slow-log-file` nor `log-file` are set, all logs are output to “`stderr`”.
- Default value: “”

11.9.2.1.8 slow-log-threshold

- The threshold for outputting slow logs. If the processing time is longer than this threshold, slow logs are output.
- Default value: “1s”

11.9.2.2 server

- Configuration items related to the server

11.9.2.3 status-thread-pool-size

- The number of worker threads for the HTTP API service
- Default value: 1
- Minimum value: 1

11.9.2.3.1 grpc-compression-type

- The compression algorithm for gRPC messages
- Optional values: “`none`”, “`deflate`”, “`gzip`”
- Default value: “`none`”
- Note: When the value is `gzip`, TiDB Dashboard will have a display error because it might not complete the corresponding compression algorithm in some cases. If you adjust the value back to the default `none`, TiDB Dashboard will display normally.

11.9.2.3.2 grpc-concurrency

- The number of gRPC worker threads. When you modify the size of the gRPC thread pool, refer to [Performance tuning for TiKV thread pools](#).
- Default value: 5
- Minimum value: 1

11.9.2.3.3 `grpc-concurrent-stream`

- The maximum number of concurrent requests allowed in a gRPC stream
- Default value: 1024
- Minimum value: 1

11.9.2.3.4 `grpc-memory-pool-quota`

- Limits the memory size that can be used by gRPC
- Default: No limit
- Limit the memory in case OOM is observed. Note that limit the usage can lead to potential stall

11.9.2.3.5 `grpc-raft-conn-num`

- The maximum number of links among TiKV nodes for Raft communication
- Default: 1
- Minimum value: 1

11.9.2.3.6 `max-grpc-send-msg-len`

- Sets the maximum length of a gRPC message that can be sent
- Default: 10485760
- Unit: Bytes
- Maximum value: 2147483647

11.9.2.3.7 `grpc-stream-initial-window-size`

- The window size of the gRPC stream
- Default: 2MB
- Unit: KB|MB|GB
- Minimum value: "1KB"

11.9.2.3.8 `grpc-keepalive-time`

- The time interval at which that gRPC sends keepalive Ping messages
- Default: "10s"
- Minimum value: "1s"

11.9.2.3.9 `grpc-keepalive-timeout`

- Disables the timeout for gRPC streams
- Default: "3s"
- Minimum value: "1s"

11.9.2.3.10 `concurrent-send-snap-limit`

- The maximum number of snapshots sent at the same time
- Default value: 32
- Minimum value: 1

11.9.2.3.11 `concurrent-recv-snap-limit`

- The maximum number of snapshots received at the same time
- Default value: 32
- Minimum value: 1

11.9.2.3.12 `end-point-recursion-limit`

- The maximum number of recursive levels allowed when TiKV decodes the Coprocessor DAG expression
- Default value: 1000
- Minimum value: 1

11.9.2.3.13 `end-point-request-max-handle-duration`

- The longest duration allowed for a TiDB's push down request to TiKV for processing tasks
- Default value: "60s"
- Minimum value: "1s"

11.9.2.3.14 `snap-max-write-bytes-per-sec`

- The maximum allowable disk bandwidth when processing snapshots
- Default value: "100MB"
- Unit: KB|MB|GB
- Minimum value: "1KB"

11.9.2.3.15 `end-point-slow-log-threshold`

- The time threshold for a TiDB's push-down request to output slow log. If the processing time is longer than this threshold, the slow logs are output.
- Default value: "1s"
- Minimum value: 0

11.9.2.3.16 `raft-client-queue-size`

- Specifies the queue size of the Raft messages in TiKV. If too many messages not sent in time result in a full buffer, or messages discarded, you can specify a greater value to improve system stability.
- Default value: 8192

11.9.2.4 `readpool.unified`

Configuration items related to the single thread pool serving read requests. This thread pool supersedes the original storage thread pool and coprocessor thread pool since the 4.0 version.

11.9.2.4.1 `min-thread-count`

- The minimal working thread count of the unified read pool
- Default value: 1

11.9.2.4.2 `max-thread-count`

- The maximum working thread count of the unified read pool or the UnifyReadPool thread pool. When you modify the size of this thread pool, refer to [Performance tuning for TiKV thread pools](#).
- Default value: MAX(4, CPU * 0.8)

11.9.2.4.3 `stack-size`

- The stack size of the threads in the unified thread pool
- Type: Integer + Unit
- Default value: "10MB"
- Unit: KB|MB|GB
- Minimum value: "2MB"
- Maximum value: The number of Kbytes output in the result of the `ulimit -sH` command executed in the system.

11.9.2.4.4 max-tasks-per-worker

- The maximum number of tasks allowed for a single thread in the unified read pool. `Server Is Busy` is returned when the value is exceeded.
- Default value: 2000
- Minimum value: 2

11.9.2.5 readpool.storage

Configuration items related to storage thread pool.

11.9.2.5.1 use-unified-pool

- Determines whether to use the unified thread pool (configured in `readpool.unified`) for storage requests. If the value of this parameter is `false`, a separate thread pool is used, which is configured through the rest parameters in this section (`readpool.→ storage`).
- Default value: If this section (`readpool.storage`) has no other configurations, the default value is `true`. Otherwise, for the backward compatibility, the default value is `false`. Change the configuration in `readpool.unified` as needed before enabling this option.

11.9.2.5.2 high-concurrency

- The allowable number of concurrent threads that handle high-priority `read` requests
- When `8 <= cpu_num < 16`, the default value is `cpu_num * 0.5`; when `cpu_num` is greater than 8, the default value is 4; when `cpu_num` is greater than 16, the default value is 8.
- Minimum value: 1

11.9.2.5.3 normal-concurrency

- The allowable number of concurrent threads that handle normal-priority `read` requests
- When `8 <= cpu_num < 16`, the default value is `cpu_num * 0.5`; when `cpu_num` is greater than 8, the default value is 4; when `cpu_num` is greater than 16, the default value is 8.
- Minimum value: 1

11.9.2.5.4 low-concurrency

- The allowable number of concurrent threads that handle low-priority `read` requests
- When `8 <= cpu_num < 16`, the default value is `cpu_num * 0.5`; when `cpu_num` is greater than 8, the default value is 4; when `cpu_num` is greater than 16, the default value is 8.
- Minimum value: 1

11.9.2.5.5 max-tasks-per-worker-high

- The maximum number of tasks allowed for a single thread in a high-priority thread pool. `Server Is Busy` is returned when the value is exceeded.
- Default value: 2000
- Minimum value: 2

11.9.2.5.6 max-tasks-per-worker-normal

- The maximum number of tasks allowed for a single thread in a normal-priority thread pool. `Server Is Busy` is returned when the value is exceeded.
- Default value: 2000
- Minimum value: 2

11.9.2.5.7 max-tasks-per-worker-low

- The maximum number of tasks allowed for a single thread in a low-priority thread pool. `Server Is Busy` is returned when the value is exceeded.
- Default value: 2000
- Minimum value: 2

11.9.2.5.8 stack-size

- The stack size of threads in the Storage read thread pool
- Type: Integer + Unit
- Default value: "10MB"
- Unit: KB|MB|GB
- Minimum value: "2MB"
- Maximum value: The number of Kbytes output in the result of the `ulimit -sH` command executed in the system.

11.9.2.6 readpool.coprocessor

Configuration items related to the Coprocessor thread pool.

11.9.2.6.1 use-unified-pool

- Determines whether to use the unified thread pool (configured in `readpool.unified` →) for coprocessor requests. If the value of this parameter is `false`, a separate thread pool is used, which is configured through the rest parameters in this section (`readpool.coprocessor`).
- Default value: If none of the parameters in this section (`readpool.coprocessor`) are set, the default value is `true`. Otherwise, the default value is `false` for the backward compatibility. Adjust the configuration items in `readpool.unified` before enabling this parameter.

11.9.2.6.2 `high-concurrency`

- The allowable number of concurrent threads that handle high-priority Coprocessor requests, such as checkpoints
- Default value: `CPU * 0.8`
- Minimum value: 1

11.9.2.6.3 `normal-concurrency`

- The allowable number of concurrent threads that handle normal-priority Coprocessor requests
- Default value: `CPU * 0.8`
- Minimum value: 1

11.9.2.6.4 `low-concurrency`

- The allowable number of concurrent threads that handle low-priority Coprocessor requests, such as table scan
- Default value: `CPU * 0.8`
- Minimum value: 1

11.9.2.6.5 `max-tasks-per-worker-high`

- The number of tasks allowed for a single thread in a high-priority thread pool. When this number is exceeded, `Server Is Busy` is returned.
- Default value: 2000
- Minimum value: 2

11.9.2.6.6 `max-tasks-per-worker-normal`

- The number of tasks allowed for a single thread in a normal-priority thread pool. When this number is exceeded, `Server Is Busy` is returned.
- Default value: 2000
- Minimum value: 2

11.9.2.6.7 `max-tasks-per-worker-low`

- The number of tasks allowed for a single thread in a low-priority thread pool. When this number is exceeded, `Server Is Busy` is returned.
- Default value: 2000
- Minimum value: 2

11.9.2.6.8 stack-size

- The stack size of the thread in the Coprocessor thread pool
- Type: Integer + Unit
- Default value: "10MB"
- Unit: KB|MB|GB
- Minimum value: "2MB"
- Maximum value: The number of Kbytes output in the result of the `ulimit -sH` command executed in the system.

11.9.2.7 storage

Configuration items related to storage

11.9.2.7.1 scheduler-concurrency

- A built-in memory lock mechanism to prevent simultaneous operations on a key. Each key has a hash in a different slot.
- Default value: 524288
- Minimum value: 1

11.9.2.7.2 scheduler-worker-pool-size

- The number of `scheduler` threads, mainly used for checking transaction consistency before data writing. If the number of CPU cores is greater than or equal to 16, the default value is 8; otherwise, the default value is 4. When you modify the size of the Scheduler thread pool, refer to [Performance tuning for TiKV thread pools](#).
- Default value: 4
- Minimum value: 1

11.9.2.7.3 scheduler-pending-write-threshold

- The maximum size of the write queue. A `Server Is Busy` error is returned for a new write to TiKV when this value is exceeded.
- Default value: "100MB"
- Unit: MB|GB

11.9.2.7.4 reserve-space

- When TiKV is started, some space is reserved on the disk as disk protection. When the remaining disk space is less than the reserved space, TiKV restricts some write operations. The reserved space is divided into two parts: 80% of the reserved space is

used as the extra disk space required for operations when the disk space is insufficient, and the other 20% is used to store the temporary file. In the process of reclaiming space, if the storage is exhausted by using too much extra disk space, this temporary file serves as the last protection for restoring services.

- The name of the temporary file is `space_placeholder_file`, located in the `storage`. ↵ `data-dir` directory. When TiKV goes offline because its disk space ran out, if you restart TiKV, the temporary file is automatically deleted and TiKV tries to reclaim the space.
- When the remaining space is insufficient, TiKV does not create the temporary file. The effectiveness of the protection is related to the size of the reserved space. The size of the reserved space is the larger value between 5% of the disk capacity and this configuration value. When the value of this configuration item is "0MB", TiKV disables this disk protection feature.
- Default value: "5GB"
- Unite: MB|GB

11.9.2.7.5 enable-ttl

- TTL is short for “Time to live”. If this item is enabled, TiKV automatically deletes data that reaches its TTL. To set the value of TTL, you need to specify it in the requests when writing data via the client. If the TTL is not specified, it means that TiKV does not automatically delete the corresponding data.
- Note: The TTL feature is only available for the RawKV interface for now. You can only configure this feature when creating a new cluster because TTL uses different data formats in the storage layer. If you modify this item on an existing cluster, TiKV reports errors when it starts.
- Default value: `false`

11.9.2.7.6 ttl-check-poll-interval

- The interval of checking data to reclaim physical spaces. If data reaches its TTL, TiKV forcibly reclaims its physical space during the check.
- Default value: "12h"
- Minimum value: "0s"

11.9.2.8 storage.block-cache

Configuration items related to the sharing of block cache among multiple RocksDB Column Families (CF). When these configuration items are enabled, block cache separately configured for each column family is disabled.

11.9.2.8.1 shared

- Enables or disables the sharing of block cache.
- Default value: `true`

11.9.2.8.2 capacity

- The size of the shared block cache.
- Default value: 45% of the size of total system memory
- Unit: KB|MB|GB

11.9.2.9 storage.flow-control

Configuration items related to the flow control mechanism in TiKV. This mechanism replaces the write stall mechanism in RocksDB and controls flow at the scheduler layer, which avoids the issue of QPS drop caused by the stuck Raftstore or Apply threads when the write traffic is high.

11.9.2.9.1 enable

- Determines whether to enable the flow control mechanism. After it is enabled, TiKV automatically disables the write stall mechanism of KvDB and the write stall mechanism of RaftDB (excluding memtable).
- Default value: true

11.9.2.9.2 memtables-threshold

- When the number of kvDB memtables reaches this threshold, the flow control mechanism starts to work.
- Default value: 5

11.9.2.9.3 10-files-threshold

- When the number of kvDB L0 files reaches this threshold, the flow control mechanism starts to work.
- Default value: 20

11.9.2.9.4 soft-pending-compaction-bytes-limit

- When the pending compaction bytes in KvDB reach this threshold, the flow control mechanism starts to reject some write requests and reports the `ServerIsBusy` error.
- Default value: "192GB"

11.9.2.9.5 hard-pending-compaction-bytes-limit

- When the pending compaction bytes in KvDB reach this threshold, the flow control mechanism rejects all write requests and reports the `ServerIsBusy` error.
- Default value: "1024GB"

11.9.2.10 storage.io-rate-limit

Configuration items related to the I/O rate limiter.

11.9.2.10.1 max-bytes-per-sec

- Limits the maximum I/O bytes that a server can write to or read from the disk (determined by the `mode` configuration item below) in one second. When this limit is reached, TiKV prefers throttling background operations over foreground ones. The value of this configuration item should be set to the disk's optimal I/O bandwidth, for example, the maximum I/O bandwidth specified by your cloud disk vendor. When this configuration value is set to zero, disk I/O operations are not limited.
- Default value: "0MB"

11.9.2.10.2 mode

- Determines which types of I/O operations are counted and restrained below the `max-bytes-per-sec` threshold. Currently, only the write-only mode is supported.
- Optional value: "write-only"
- Default value: "write-only"

11.9.2.11 raftstore

Configuration items related to Raftstore

11.9.2.11.1 prevote

- Enables or disables `prevote`. Enabling this feature helps reduce jitter on the system after recovery from network partition.
- Default value: `true`

11.9.2.11.2 raftdb-path

- The path to the Raft library, which is `storage.data-dir/raft` by default
- Default value: ""

11.9.2.11.3 raft-base-tick-interval

- The time interval at which the Raft state machine ticks
- Default value: "1s"
- Minimum value: greater than 0

11.9.2.11.4 `raft-heartbeat-ticks`

- The number of passed ticks when the heartbeat is sent. This means that a heartbeat is sent at the time interval of `raft-base-tick-interval * raft-heartbeat-ticks`.
- Default value: 2
- Minimum value: greater than 0

11.9.2.11.5 `raft-election-timeout-ticks`

- The number of passed ticks when Raft election is initiated. This means that if Raft group is missing the leader, a leader election is initiated approximately after the time interval of `raft-base-tick-interval * raft-election-timeout-ticks`.
- Default value: 10
- Minimum value: `raft-heartbeat-ticks`

11.9.2.11.6 `raft-min-election-timeout-ticks`

- The minimum number of ticks during which the Raft election is initiated. If the number is 0, the value of `raft-election-timeout-ticks` is used. The value of this parameter must be greater than or equal to `raft-election-timeout-ticks`.
- Default value: 0
- Minimum value: 0

11.9.2.11.7 `raft-max-election-timeout-ticks`

- The maximum number of ticks during which the Raft election is initiated. If the number is 0, the value of `raft-election-timeout-ticks * 2` is used.
- Default value: 0
- Minimum value: 0

11.9.2.11.8 `raft-max-size-per-msg`

- The soft limit on the size of a single message packet
- Default value: "1MB"
- Minimum value: 0
- Unit: MB

11.9.2.11.9 `raft-max-inflight-msgs`

- The number of Raft logs to be confirmed. If this number is exceeded, log sending slows down.
- Default value: 256
- Minimum value: greater than 0

11.9.2.11.10 `raft-entry-max-size`

- The hard limit on the maximum size of a single log
- Default value: "8MB"
- Minimum value: 0
- Unit: MB|GB

11.9.2.11.11 `raft-log-gc-tick-interval`

- The time interval at which the polling task of deleting Raft logs is scheduled. 0 means that this feature is disabled.
- Default value: "10s"
- Minimum value: 0

11.9.2.11.12 `raft-log-gc-threshold`

- The soft limit on the maximum allowable count of residual Raft logs
- Default value: 50
- Minimum value: 1

11.9.2.11.13 `raft-log-gc-count-limit`

- The hard limit on the allowable number of residual Raft logs
- Default value: the log number that can be accommodated in the 3/4 Region size (calculated as 1MB for each log)
- Minimum value: 0

11.9.2.11.14 `raft-log-gc-size-limit`

- The hard limit on the allowable size of residual Raft logs
- Default value: 3/4 of the Region size
- Minimum value: greater than 0

11.9.2.11.15 `raft-entry-cache-life-time`

- The maximum remaining time allowed for the log cache in memory.
- Default value: "30s"
- Minimum value: 0

11.9.2.11.16 `hibernate-regions`

- Enables or disables Hibernate Region. When this option is enabled, a Region idle for a long time is automatically set as hibernated. This reduces the extra overhead caused by heartbeat messages between the Raft leader and the followers for idle Regions. You can use `peer-stale-state-check-interval` to modify the heartbeat interval between the leader and the followers of hibernated Regions.
- Default value: `true` in v5.0.2 and later versions; `false` in versions before v5.0.2

11.9.2.11.17 `split-region-check-tick-interval`

- Specifies the interval at which to check whether the Region split is needed. 0 means that this feature is disabled.
- Default value: "10s"
- Minimum value: 0

11.9.2.11.18 `region-split-check-diff`

- The maximum value by which the Region data is allowed to exceed before Region split
- Default value: 1/16 of the Region size.
- Minimum value: 0

11.9.2.11.19 `region-compact-check-interval`

- The time interval at which to check whether it is necessary to manually trigger RocksDB compaction. 0 means that this feature is disabled.
- Default value: "5m"
- Minimum value: 0

11.9.2.11.20 `region-compact-check-step`

- The number of Regions checked at one time for each round of manual compaction
- Default value: 100
- Minimum value: 0

11.9.2.11.21 `region-compact-min-tombstones`

- The number of tombstones required to trigger RocksDB compaction
- Default value: 10000
- Minimum value: 0

11.9.2.11.22 `region-compact-tombstones-percent`

- The proportion of tombstone required to trigger RocksDB compaction
- Default value: 30
- Minimum value: 1
- Maximum value: 100

11.9.2.11.23 `pd-heartbeat-tick-interval`

- The time interval at which a Region's heartbeat to PD is triggered. 0 means that this feature is disabled.
- Default value: "1m"
- Minimum value: 0

11.9.2.11.24 `pd-store-heartbeat-tick-interval`

- The time interval at which a store's heartbeat to PD is triggered. 0 means that this feature is disabled.
- Default value: "10s"
- Minimum value: 0

11.9.2.11.25 `snap-mgr-gc-tick-interval`

- The time interval at which the recycle of expired snapshot files is triggered. 0 means that this feature is disabled.
- Default value: "1m"
- Minimum value: 0

11.9.2.11.26 `snap-gc-timeout`

- The longest time for which a snapshot file is saved
- Default value: "4h"
- Minimum value: 0

11.9.2.11.27 `lock-cf-compact-interval`

- The time interval at which TiKV triggers a manual compaction for the Lock Column Family
- Default value: "256MB"
- Default value: "10m"
- Minimum value: 0

11.9.2.11.28 lock-cf-compact-bytes-threshold

- The size out of which TiKV triggers a manual compaction for the Lock Column Family
- Default value: "256MB"
- Minimum value: 0
- Unit: MB

11.9.2.11.29 notify-capacity

- The longest length of the Region message queue.
- Default value: 40960
- Minimum value: 0

11.9.2.11.30 messages-per-tick

- The maximum number of messages processed per batch
- Default value: 4096
- Minimum value: 0

11.9.2.11.31 max-peer-down-duration

- The longest inactive duration allowed for a peer. A peer with timeout is marked as down, and PD tries to delete it later.
- Default value: "10m"
- Minimum value: When Hibernate Region is enabled, the minimum value is peer-stale ↵ -check-interval * 2; when Hibernate Region is disabled, the minimum value is 0.

11.9.2.11.32 max-leader-missing-duration

- The longest duration allowed for a peer to be in the state where a Raft group is missing the leader. If this value is exceeded, the peer verifies with PD whether the peer has been deleted.
- Default value: "2h"
- Minimum value: greater than abnormal-leader-missing-duration

11.9.2.11.33 abnormal-leader-missing-duration

- The longest duration allowed for a peer to be in the state where a Raft group is missing the leader. If this value is exceeded, the peer is seen as abnormal and marked in metrics and logs.
- Default value: "10m"
- Minimum value: greater than peer-stale-state-check-interval

11.9.2.11.34 peer-stale-state-check-interval

- The time interval to trigger the check for whether a peer is in the state where a Raft group is missing the leader.
- Default value: "5m"
- Minimum value: greater than `2 * election-timeout`

11.9.2.11.35 leader-transfer-max-log-lag

- The maximum number of missing logs allowed for the transferee during a Raft leader transfer
- Default value: 128
- Minimum value: 10

11.9.2.11.36 snap-apply-batch-size

- The memory cache size required when the imported snapshot file is written into the disk
- Default value: "10MB"
- Minimum value: 0
- Unit: MB

11.9.2.11.37 consistency-check-interval

- The time interval at which the consistency check is triggered. 0 means that this feature is disabled.
- Default value: "0s"
- Minimum value: 0

11.9.2.11.38 raft-store-max-leader-lease

- The longest trusted period of a Raft leader
- Default value: "9s"
- Minimum value: 0

11.9.2.11.39 allow-remove-leader

- Determines whether to allow deleting the main switch
- Default value: `false`

11.9.2.11.40 `merge-max-log-gap`

- The maximum number of missing logs allowed when `merge` is performed
- Default value: 10
- Minimum value: greater than `raft-log-gc-count-limit`

11.9.2.11.41 `merge-check-tick-interval`

- The time interval at which TiKV checks whether a Region needs merge
- Default value: "2s"
- Minimum value: greater than 0

11.9.2.11.42 `use-delete-range`

- Determines whether to delete data from the `rocksdb delete_range` interface
- Default value: `false`

11.9.2.11.43 `cleanup-import-sst-interval`

- The time interval at which the expired SST file is checked. 0 means that this feature is disabled.
- Default value: "10m"
- Minimum value: 0

11.9.2.11.44 `local-read-batch-size`

- The maximum number of read requests processed in one batch
- Default value: 1024
- Minimum value: greater than 0

11.9.2.11.45 `apply-max-batch-size`

- The maximum number of requests for data flushing in one batch
- Default value: 256
- Minimum value: greater than 0

11.9.2.11.46 `apply-pool-size`

- The allowable number of threads in the pool that flushes data to storage. When you modify the size of this thread pool, refer to [Performance tuning for TiKV thread pools](#).
- Default value: 2
- Minimum value: greater than 0

11.9.2.11.47 `store-max-batch-size`

- The maximum number of requests processed in one batch
- If `hibernate-regions` is enabled, the default value is 256. If `hibernate-regions` is disabled, the default value is 1024.
- Minimum value: greater than 0

11.9.2.11.48 `store-pool-size`

- The allowable number of threads that process Raft, which is the size of the Raftstore thread pool. When you modify the size of this thread pool, refer to [Performance tuning for TiKV thread pools](#).
- Default value: 2
- Minimum value: greater than 0

11.9.2.11.49 `store-io-pool-size` New in v5.3.0

- The allowable number of threads that process Raft I/O tasks, which is the size of the StoreWriter thread pool. When you modify the size of this thread pool, refer to [Performance tuning for TiKV thread pools](#).
- Default value: 0
- Minimum value: 0

11.9.2.11.50 `future-poll-size`

- The allowable number of threads that drive `future`
- Default value: 1
- Minimum value: greater than 0

11.9.2.11.51 `cmd-batch`

- Controls whether to enable batch processing of the requests. When it is enabled, the write performance is significantly improved.
- Default value: `true`

11.9.2.11.52 `inspect-interval`

- At a certain interval, TiKV inspects the latency of the Raftstore component. This parameter specifies the interval of the inspection. If the latency exceeds this value, this inspection is marked as timeout.
- Judges whether the TiKV node is slow based on the ratio of timeout inspection.
- Default value: "500ms"
- Minimum value: "1ms"

11.9.2.11.53 `raft-write-size-limit` New in v5.3.0

- Determines the threshold at which Raft data is written into the disk. If the data size is larger than the value of this configuration item, the data is written to the disk. When the value of `store-io-pool-size` is 0, this configuration item does not take effect.
- Default value: 1MB
- Minimum value: 0

11.9.2.11.54 `raft-msg-flush-interval` New in v5.3.0

- Determines the interval at which Raft messages are sent in batches. The Raft messages in batches are sent at every interval specified by this configuration item. When the value of `store-io-pool-size` is 0, this configuration item does not take effect.
- Default value: 250us
- Minimum value: 0

11.9.2.12 Coprocessor

Configuration items related to Coprocessor

11.9.2.12.1 `split-region-on-table`

- Determines whether to split Region by table. It is recommended for you to use the feature only in TiDB mode.
- Default value: false

11.9.2.12.2 `batch-split-limit`

- The threshold of Region split in batches. Increasing this value speeds up Region split.
- Default value: 10
- Minimum value: 1

11.9.2.12.3 `region-max-size`

- The maximum size of a Region. When the value is exceeded, the Region splits into many.
- Default value: "144MB"
- Unit: KB|MB|GB

11.9.2.12.4 `region-split-size`

- The size of the newly split Region. This value is an estimate.
- Default value: "96MB"
- Unit: KB|MB|GB

11.9.2.12.5 `region-max-keys`

- The maximum allowable number of keys in a Region. When this value is exceeded, the Region splits into many.
- Default value: 1440000

11.9.2.12.6 `region-split-keys`

- The number of keys in the newly split Region. This value is an estimate.
- Default value: 960000

11.9.2.13 RocksDB

Configuration items related to RocksDB

11.9.2.13.1 `max-background-jobs`

- The number of background threads in RocksDB. When you modify the size of the RocksDB thread pool, refer to [Performance tuning for TiKV thread pools](#).
- Default value: 8
- Minimum value: 2

11.9.2.13.2 `max-background-flushes`

- The maximum number of concurrent background memtable flush jobs
- Default value: 2
- Minimum value: 1

11.9.2.13.3 `max-sub-compactions`

- The number of sub-compaction operations performed concurrently in RocksDB
- Default value: 3
- Minimum value: 1

11.9.2.13.4 `max-open-files`

- The total number of files that RocksDB can open
- Default value: 40960
- Minimum value: -1

11.9.2.13.5 `max-manifest-file-size`

- The maximum size of a RocksDB Manifest file
- Default value: "128MB"
- Minimum value: 0
- Unit: B|KB|MB|GB

11.9.2.13.6 `create-if-missing`

- Determines whether to automatically create a DB switch
- Default value: `true`

11.9.2.13.7 `wal-recovery-mode`

- WAL recovery mode
- Value options: 0, 1, 2, 3
- 0 (`TolerateCorruptedTailRecords`): tolerates and discards the records that have incomplete trailing data on all logs.
- 1 (`AbsoluteConsistency`): abandons recovery when corrupted logs are found.
- 2 (`PointInTimeRecovery`): recovers log sequentially until the first corrupted log is encountered.
- 3 (`SkipAnyCorruptedRecords`): recovery after a disaster. Corrupted records are skipped
- Default value: 2
- Minimum value: 0
- Maximum value: 3

11.9.2.13.8 `wal-dir`

- The directory in which WAL files are stored
- Default value: "/tmp/tikv/store"

11.9.2.13.9 `wal-ttl-seconds`

- The living time of the archived WAL files. When the value is exceeded, the system deletes these files.
- Default value: 0
- Minimum value: 0
- unit: second

11.9.2.13.10 wal-size-limit

- The size limit of the archived WAL files. When the value is exceeded, the system deletes these files.
- Default value: 0
- Minimum value: 0
- Unit: B|KB|MB|GB

11.9.2.13.11 enable-statistics

- Determines whether to enable the statistics of RocksDB
- Default value: true

11.9.2.13.12 stats-dump-period

- The interval at which statistics are output to the log.
- Default value: 10m

11.9.2.13.13 compaction-readahead-size

- Enables the readahead feature during RocksDB compaction and specifies the size of readahead data. If you are using mechanical disks, it is recommended to set the value to 2MB at least.
- Default value: 0
- Minimum value: 0
- Unit: B|KB|MB|GB

11.9.2.13.14 writable-file-max-buffer-size

- The maximum buffer size used in WritableFileWrite
- Default value: "1MB"
- Minimum value: 0
- Unit: B|KB|MB|GB

11.9.2.13.15 use-direct-io-for-flush-and-compaction

- Determines whether to use O_DIRECT for both reads and writes in the background flush and compactions. The performance impact of this option: enabling O_DIRECT bypasses and prevents contamination of the OS buffer cache, but the subsequent file reads require re-reading the contents to the buffer cache.
- Default value: false

11.9.2.13.16 `rate-bytes-per-sec`

- The maximum rate permitted by RocksDB's compaction rate limiter
- Default value: 10GB
- Minimum value: 0
- Unit: B|KB|MB|GB

11.9.2.13.17 `rate-limiter-mode`

- RocksDB's compaction rate limiter mode
- Optional values: 1 (`ReadOnly`), 2 (`WriteOnly`), 3 (`AllIo`)
- Default value: 2
- Minimum value: 1
- Maximum value: 3

11.9.2.13.18 `rate-limiter-auto-tuned` New in v5.0

- Determines whether to automatically optimize the configuration of the RocksDB's compaction rate limiter based on recent workload. When this configuration is enabled, compaction pending bytes will be slightly higher than usual.
- Default value: `true`

11.9.2.13.19 `enable-pipelined-write`

- Enables or disables Pipelined Write
- Default value: `true`

11.9.2.13.20 `bytes-per-sync`

- The rate at which OS incrementally synchronizes files to disk while these files are being written asynchronously
- Default value: "1MB"
- Minimum value: 0
- Unit: B|KB|MB|GB

11.9.2.13.21 `wal-bytes-per-sync`

- The rate at which OS incrementally synchronizes WAL files to disk while the WAL files are being written
- Default value: "512KB"
- Minimum value: 0
- Unit: B|KB|MB|GB

11.9.2.13.22 `info-log-max-size`

- The maximum size of Info log
- Default value: "1GB"
- Minimum value: 0
- Unit: B|KB|MB|GB

11.9.2.13.23 `info-log-roll-time`

- The time interval at which Info logs are truncated. If the value is 0s, logs are not truncated.
- Default value: "0s"

11.9.2.13.24 `info-log-keep-log-file-num`

- The maximum number of kept log files
- Default value: 10
- Minimum value: 0

11.9.2.13.25 `info-log-dir`

- The directory in which logs are stored
- Default value: “”

11.9.2.14 `rocksdb.titan`

Configuration items related to Titan

11.9.2.14.1 `enabled`

- Enables or disables Titan
- Default value: `false`

11.9.2.14.2 `dirname`

- The directory in which the Titan Blob file is stored
- Default value: "titandb"

11.9.2.14.3 `disable-gc`

- Determines whether to disable Garbage Collection (GC) that Titan performs to Blob files
- Default value: `false`

11.9.2.14.4 `max-background-gc`

- The maximum number of GC threads in Titan
- Default value: 4
- Minimum value: 1

11.9.2.15 `rocksdb.defaultcf` | `rocksdb.writecf` | `rocksdb.lockcf`

Configuration items related to `rocksdb.defaultcf`, `rocksdb.writecf`, and `rocksdb.lockcf`.

11.9.2.15.1 `block-size`

- The default size of a RocksDB block
- Default value for `defaultcf` and `writecf`: "64KB"
- Default value for `lockcf`: "16KB"
- Minimum value: "1KB"
- Unit: KB|MB|GB

11.9.2.15.2 `block-cache-size`

- The cache size of a RocksDB block
- Default value for `defaultcf`: Total machine memory * 25%
- Default value for `writecf`: Total machine memory * 15%
- Default value for `lockcf`: Total machine memory * 2%
- Minimum value: 0
- Unit: KB|MB|GB

11.9.2.15.3 `disable-block-cache`

- Enables or disables block cache
- Default value: `false`

11.9.2.15.4 `cache-index-and-filter-blocks`

- Enables or disables caching index and filter
- Default value: `true`

11.9.2.15.5 `pin-l0-filter-and-index-blocks`

- Determines whether to pin the index and filter blocks of the level 0 SST files in memory.
- Default value: `true`

11.9.2.15.6 `use-bloom-filter`

- Enables or disables bloom filter
- Default value: `true`

11.9.2.15.7 `optimize-filters-for-hits`

- Determines whether to optimize the hit ratio of filters
- Default value for `defaultcf`: `true`
- Default value for `writetcf` and `lockcf`: `false`

11.9.2.15.8 `whole-key-filtering`

- Determines whether to put the entire key to bloom filter
- Default value for `defaultcf` and `lockcf`: `true`
- Default value for `writetcf`: `false`

11.9.2.15.9 `bloom-filter-bits-per-key`

- The length that bloom filter reserves for each key
- Default value: 10
- unit: byte

11.9.2.15.10 `block-based-bloom-filter`

- Determines whether each block creates a bloom filter
- Default value: `false`

11.9.2.15.11 `read-amp-bytes-per-bit`

- Enables or disables statistics of read amplification.
- Optional values: 0 (disabled), > 0 (enabled).
- Default value: 0
- Minimum value: 0

11.9.2.15.12 `compression-per-level`

- The default compression algorithm for each level
- Optional values: `["no", "no", "lz4", "lz4", "lz4", "zstd", "zstd"]`
- Default value for `defaultcf` and `writetcf`: `["no", "no", "lz4", "lz4", "lz4", "zstd", "zstd"]`
- Default value for `lockcf`: `["no", "no", "no", "no", "no", "no", "no"]`

11.9.2.15.13 `bottommost-level-compression`

- Sets the compression algorithm of the bottommost layer. This configuration item overrides the `compression-per-level` setting.
- Ever since data is written to LSM-tree, RocksDB does not directly adopt the last compression algorithm specified in the `compression-per-level` array for the bottommost layer. `bottommost-level-compression` enables the bottommost layer to use the compression algorithm of the best compression effect from the beginning.
- If you do not want to set the compression algorithm for the bottommost layer, set the value of this configuration item to `disable`.
- Default value: "zstd"

11.9.2.15.14 `write-buffer-size`

- Memtable size
- Default value for `defaultcf` and `writetcf`: "128MB"
- Default value for `lockcf`: "32MB"
- Minimum value: 0
- Unit: KB|MB|GB

11.9.2.15.15 `max-write-buffer-number`

- The maximum number of memtables
- Default value: 5
- Minimum value: 0

11.9.2.15.16 `min-write-buffer-number-to-merge`

- The minimum number of memtables required to trigger flush
- Default value: 1
- Minimum value: 0

11.9.2.15.17 `max-bytes-for-level-base`

- The maximum number of bytes at base level (L1). Generally, it is set to 4 times the size of a memtable.
- Default value for `defaultcf` and `writetcf`: "512MB"
- Default value for `lockcf`: "128MB"
- Minimum value: 0
- Unit: KB|MB|GB

11.9.2.15.18 target-file-size-base

- The size of the target file at base level. This value is overridden by `compaction-guard` ↪ `-max-output-file-size` when the `enable-compaction-guard` value is true.
- Default: "8MB"
- Minimum value: 0
- Unit: KB|MB|GB

11.9.2.15.19 level0-file-num-compaction-trigger

- The maximum number of files at L0 that trigger compaction
- Default value for `defaultcf` and `writecf`: 4
- Default value for `lockcf`: 1
- Minimum value: 0

11.9.2.15.20 level0-slowdown-writes-trigger

- The maximum number of files at L0 that trigger write stall
- Default value: 20
- Minimum value: 0

11.9.2.15.21 level0-stop-writes-trigger

- The maximum number of files at L0 required to completely block write
- Default value: 36
- Minimum value: 0

11.9.2.15.22 max-compaction-bytes

- The maximum number of bytes written into disk per compaction
- Default value: "2GB"
- Minimum value: 0
- Unit: KB|MB|GB

11.9.2.15.23 compaction-pri

- The priority type of compaction
- Optional values: 0 (`ByCompensatedSize`), 1 (`OldestLargestSeqFirst`), 2 (`OldestSmallestSeqFirst`), 3 (`MinOverlappingRatio`)
- Default value for `defaultcf` and `writecf`: 3
- Default value for `lockcf`: 0

11.9.2.15.24 `dynamic-level-bytes`

- Determines whether to optimize dynamic level bytes
- Default value: `true`

11.9.2.15.25 `num-levels`

- The maximum number of levels in a RocksDB file
- Default value: 7

11.9.2.15.26 `max-bytes-for-level-multiplier`

- The default amplification multiple for each layer
- Default value: 10

11.9.2.15.27 `compaction-style`

- Compaction method
- Optional values: "level", "universal"
- Default value: "level"

11.9.2.15.28 `disable-auto-compactions`

- Enables or disables automatic compaction
- Default value: `false`

11.9.2.15.29 `soft-pending-compaction-bytes-limit`

- The soft limit on the pending compaction bytes
- Default value: "192GB"
- Unit: KB|MB|GB

11.9.2.15.30 `hard-pending-compaction-bytes-limit`

- The hard limit on the pending compaction bytes
- Default value: "256GB"
- Unit: KB|MB|GB

11.9.2.15.31 enable-compaction-guard

- Enables or disables the compaction guard, which is an optimization to split SST files at TiKV Region boundaries. This optimization can help reduce compaction I/O and allows TiKV to use larger SST file size (thus less SST files overall) and at the time efficiently clean up stale data when migrating Regions.
- Default value for `defaultcf` and `writecf`: `true`
- Default value for `lockcf`: `false`

11.9.2.15.32 compaction-guard-min-output-file-size

- The minimum SST file size when the compaction guard is enabled. This configuration prevents SST files from being too small when the compaction guard is enabled.
- Default value: `"8MB"`
- Unit: KB|MB|GB

11.9.2.15.33 compaction-guard-max-output-file-size

- The maximum SST file size when the compaction guard is enabled. The configuration prevents SST files from being too large when the compaction guard is enabled. This configuration overrides `target-file-size-base` for the same column family.
- Default value: `"128MB"`
- Unit: KB|MB|GB

11.9.2.16 rocksdb.defaultcf.titan

Configuration items related to `rocksdb.defaultcf.titan`.

11.9.2.16.1 min-blob-size

- The smallest value stored in a Blob file. Values smaller than the specified size are stored in the LSM-Tree.
- Default value: `"1KB"`
- Minimum value: 0
- Unit: KB|MB|GB

11.9.2.16.2 blob-file-compression

- The compression algorithm used in a Blob file
- Optional values: `"no"`, `"snappy"`, `"zlib"`, `"bzip2"`, `"lz4"`, `"lz4hc"`, `"zstd"`
- Default value: `"lz4"`

11.9.2.16.3 `blob-cache-size`

- The cache size of a Blob file
- Default value: "0GB"
- Minimum value: 0
- Unit: KB|MB|GB

11.9.2.16.4 `min-gc-batch-size`

- The minimum total size of Blob files required to perform GC for one time
- Default value: "16MB"
- Minimum value: 0
- Unit: KB|MB|GB

11.9.2.16.5 `max-gc-batch-size`

- The maximum total size of Blob files allowed to perform GC for one time
- Default value: "64MB"
- Minimum value: 0
- Unit: KB|MB|GB

11.9.2.16.6 `discardable-ratio`

- The ratio at which GC is triggered for Blob files. The Blob file can be selected for GC only if the proportion of the invalid values in a Blob file exceeds this ratio.
- Default value: 0.5
- Minimum value: 0
- Maximum value: 1

11.9.2.16.7 `sample-ratio`

- The ratio of (data read from a Blob file/the entire Blob file) when sampling the file during GC
- Default value: 0.1
- Minimum value: 0
- Maximum value: 1

11.9.2.16.8 `merge-small-file-threshold`

- When the size of a Blob file is smaller than this value, the Blob file might still be selected for GC. In this situation, `discardable-ratio` is ignored.
- Default value: "8MB"
- Minimum value: 0
- Unit: KB|MB|GB

11.9.2.16.9 blob-run-mode

- Specifies the running mode of Titan.
- Optional values:
 - `normal`: Writes data to the blob file when the value size exceeds `min-blob-size`.
 - `read_only`: Refuses to write new data to the blob file, but still reads the original data from the blob file.
 - `fallback`: Writes data in the blob file back to LSM.
- Default value: `normal`

11.9.2.16.10 level-merge

- Determines whether to optimize the read performance. When `level-merge` is enabled, there is more write amplification.
- Default value: `false`

11.9.2.16.11 gc-merge-rewrite

- Determines whether to use the merge operator to write back blob indexes for Titan GC. When `gc-merge-rewrite` is enabled, it reduces the effect of Titan GC on the writes in the foreground.
- Default value: `false`

11.9.2.17 raftdb

Configuration items related to `raftdb`

11.9.2.17.1 max-background-jobs

- The number of background threads in RocksDB. When you modify the size of the RocksDB thread pool, refer to [Performance tuning for TiKV thread pools](#).
- Default value: 4
- Minimum value: 2

11.9.2.17.2 max-sub-compactions

- The number of concurrent sub-compaction operations performed in RocksDB
- Default value: 2
- Minimum value: 1

11.9.2.17.3 wal-dir

- The directory in which WAL files are stored
- Default value: "/tmp/tikv/store"

11.9.2.18 security

Configuration items related to security

11.9.2.18.1 ca-path

- The path of the CA file
- Default value: ""

11.9.2.18.2 cert-path

- The path of the Privacy Enhanced Mail (PEM) file that contains the X.509 certificate
- Default value: ""

11.9.2.18.3 key-path

- The path of the PEM file that contains the X.509 key
- Default value: ""

11.9.2.18.4 cert-allowed-cn

- A list of acceptable X.509 Common Names in certificates presented by clients. Requests are permitted only when the presented Common Name is an exact match with one of the entries in the list.
- Default value: []. This means that the client certificate CN check is disabled by default.

11.9.2.18.5 redact-info-log New in v4.0.8

- This configuration item enables or disables log redaction. If the configuration value is set to `true`, all user data in the log will be replaced by ?.
- Default value: `false`

11.9.2.19 security.encryption

Configuration items related to [encryption at rest](#) (TDE).

11.9.2.19.1 `data-encryption-method`

- The encryption method for data files
- Value options: “plaintext”, “aes128-ctr”, “aes192-ctr”, and “aes256-ctr”
- A value other than “plaintext” means that encryption is enabled, in which case the master key must be specified.
- Default value: "plaintext"

11.9.2.19.2 `data-key-rotation-period`

- Specifies how often TiKV rotates the data encryption key.
- Default value: 7d

11.9.2.19.3 `enable-file-dictionary-log`

- Enables the optimization to reduce I/O and mutex contention when TiKV manages the encryption metadata.
- To avoid possible compatibility issues when this configuration parameter is enabled (by default), see [Encryption at Rest - Compatibility between TiKV versions](#) for details.
- Default value: true

11.9.2.19.4 `master-key`

- Specifies the master key if encryption is enabled. To learn how to configure a master key, see [Encryption at Rest - Configure encryption](#).

11.9.2.19.5 `previous-master-key`

- Specifies the old master key when rotating the new master key. The configuration format is the same as that of `master-key`. To learn how to configure a master key, see [Encryption at Rest - Configure encryption](#).

11.9.2.20 `import`

Configuration items related to TiDB Lightning import and BR restore.

11.9.2.20.1 `num-threads`

- The number of threads to process RPC requests
- Default value: 8
- Minimum value: 1

11.9.2.20.2 num-import-jobs

- The number of jobs imported concurrently
- Default value: 8
- Minimum value: 1

11.9.2.21 gc

11.9.2.21.1 enable-compaction-filter New in v5.0

- Controls whether to enable the GC in Compaction Filter feature
- Default value: true

11.9.2.22 backup

Configuration items related to BR backup.

11.9.2.22.1 num-threads

- The number of worker threads to process backup
- Default value: MIN(CPU * 0.75, 32).
- Minimum value: 1

11.9.2.23 cdc

Configuration items related to TiCDC.

11.9.2.23.1 min-ts-interval

- The interval at which Resolved TS is calculated and forwarded.
- Default value: "1s"

11.9.2.23.2 old-value-cache-memory-quota

- The upper limit of memory usage by TiCDC old values.
- Default value: 512MB

11.9.2.23.3 sink-memory-quota

- The upper limit of memory usage by TiCDC data change events.
- Default value: 512MB

11.9.2.23.4 `incremental-scan-speed-limit`

- The maximum speed at which historical data is incrementally scanned.
- Default value: "128MB", which means 128 MB per second.

11.9.2.23.5 `incremental-scan-threads`

- The number of threads for the task of incrementally scanning historical data.
- Default value: 4, which means 4 threads.

11.9.2.23.6 `incremental-scan-concurrency`

- The maximum number of concurrent executions for the tasks of incrementally scanning historical data.
- Default value: 6, which means 6 tasks can be concurrent executed at most.
- Note: The value of `incremental-scan-concurrency` must be greater than or equal to that of `incremental-scan-threads`; otherwise, TiKV will report an error at startup.

11.9.2.24 `resolved-ts`

Configuration items related to maintaining the Resolved TS to serve Stale Read requests.

11.9.2.24.1 `enable`

- Determines whether to maintain the Resolved TS for all Regions.
- Default value: `true`

11.9.2.24.2 `advance-ts-interval`

- The interval at which Resolved TS is calculated and forwarded.
- Default value: "1s"

11.9.2.24.3 `scan-lock-pool-size`

- The number of threads that TiKV uses to scan the MVCC (multi-version concurrency control) lock data when initializing the Resolved TS.
- Default value: 2, which means 2 threads.

11.9.2.25 `pessimistic-txn`

For pessimistic transaction usage, refer to [TiDB Pessimistic Transaction Mode](#).

11.9.2.25.1 wait-for-lock-timeout

- The longest time that a pessimistic transaction in TiKV waits for other transactions to release the lock. If the time is out, an error is returned to TiDB, and TiDB retries to add a lock. The lock wait timeout is set by `innodb_lock_wait_timeout`.
- Default value: `"1s"`
- Minimum value: `"1ms"`

11.9.2.25.2 wait-up-delay-duration

- When pessimistic transactions release the lock, among all the transactions waiting for lock, only the transaction with the smallest `start_ts` is woken up. Other transactions will be woken up after `wait-up-delay-duration`.
- Default value: `"20ms"`

11.9.2.25.3 pipelined

- This configuration item enables the pipelined process of adding the pessimistic lock. With this feature enabled, after detecting that data can be locked, TiKV immediately notifies TiDB to execute the subsequent requests and write the pessimistic lock asynchronously, which reduces most of the latency and significantly improves the performance of pessimistic transactions. But there is a still low probability that the asynchronous write of the pessimistic lock fails, which might cause the failure of pessimistic transaction commits.
- Default value: `true`

11.9.3 Configure TiFlash

This document introduces the configuration parameters related to the deployment and use of TiFlash.

11.9.3.1 PD scheduling parameters

You can adjust the PD scheduling parameters using `pd-ctl`. Note that you can use `tiup ↳ ctl pd` to replace `pd-ctl -u <pd_ip:pd_port>` when using `tiup` to deploy and manage your cluster.

- **replica-schedule-limit**: determines the rate at which the replica-related operator is generated. The parameter affects operations such as making nodes offline and add replicas.

Notes:

The value of this parameter should be less than that of `region-schedule ↵ -limit`. Otherwise, the normal Region scheduling among TiKV nodes is affected.

- `store-balance-rate`: limits the rate at which Regions of each TiKV/TiFlash store are scheduled. Note that this parameter takes effect only when the stores have newly joined the cluster. If you want to change the setting for existing stores, use the following command.

Note:

Since v4.0.2, the `store-balance-rate` parameter has been deprecated and changes have been made to the `store limit` command. See [store-limit](#) for details.

- Execute the `pd-ctl -u <pd_ip:pd_port> store limit <store_id> <value>` command to set the scheduling rate of a specified store. (To get `store_id`, you can execute the `pd-ctl -u <pd_ip:pd_port> store` command.)
- If you do not set the scheduling rate for Regions of a specified store, this store inherits the setting of `store-balance-rate`.
- You can execute the `pd-ctl -u <pd_ip:pd_port> store limit` command to view the current setting value of `store-balance-rate`.
- `replication.location-labels`: indicates the topological relationship of TiKV instances. The order of the keys indicates the layering relationship of different labels. If TiFlash is enabled, you need to use `pd-ctl config placement-rules` to set the default value. For details, see [geo-distributed-deployment-topology](#).

11.9.3.2 TiFlash configuration parameters

This section introduces the configuration parameters of TiFlash.

11.9.3.2.1 Configure the `tiflash.toml` file

```
#### The listening host for supporting services such as TPC/HTTP. It is
    ↵ recommended to configure it as "0.0.0.0", which means to listen on
    ↵ all IP addresses of this machine.
listen_host = "0.0.0.0"
#### The TiFlash TCP service port.
tcp_port = 9000
#### The TiFlash HTTP service port.
http_port = 8123
```

```

##### The cache size limit of the metadata of a data block. Generally, you do
    ↪ not need to change this value.
mark_cache_size = 5368709120
##### The cache size limit of the min-max index of a data block. Generally,
    ↪ you do not need to change this value.
minmax_index_cache_size = 5368709120
##### The cache size limit of the DeltaIndex. The default value is 0, which
    ↪ means no limit.
delta_index_cache_size = 0

##### The storage path of TiFlash data. If there are multiple directories,
    ↪ separate each directory with a comma.
##### path and path_realtime_mode are deprecated since v4.0.9. Use the
    ↪ configurations
##### in the [storage] section to get better performance in the multi-disk
    ↪ deployment scenarios
##### Since TiDB v5.2.0, if you need to use the storage.io_rate_limit
    ↪ configuration, you need to set the storage path of TiFlash data to
    ↪ storage.main.dir at the same time.
##### When the [storage] configurations exist, both path and
    ↪ path_realtime_mode configurations are ignored.
### path = "/tidb-data/tiflash-9000"
#### or
### path = "/ssd0/tidb-data/tiflash,/ssd1/tidb-data/tiflash,/ssd2/tidb-data/
    ↪ tiflash"
##### The default value is false. If you set it to true and multiple
    ↪ directories
##### are set in the path, the latest data is stored in the first directory
    ↪ and older
##### data is stored in the rest directories.
### path_realtime_mode = false

##### The path in which the TiFlash temporary files are stored. By default it
    ↪ is the first directory in path
##### or in storage.latest.dir appended with "/tmp".
### tmp_path = "/tidb-data/tiflash-9000/tmp"

##### Storage paths settings take effect starting from v4.0.9
[storage]
## This configuration item is deprecated since v5.2.0. You can use the [
    ↪ storage.io_rate_limit] settings below instead.
# bg_task_io_rate_limit = 0

[storage.main]

```

```

## The list of directories to store the main data. More than 90% of the
## total data is stored in
## the directory list.
dir = [ "/tidb-data/tiflash-9000" ]
## or
# dir = [ "/ssd0/tidb-data/tiflash", "/ssd1/tidb-data/tiflash" ]

## The maximum storage capacity of each directory in storage.main.dir.
## If it is not set, or is set to multiple 0, the actual disk (the disk
## where the directory is located) capacity is used.
## Note that human-readable numbers such as "10GB" are not supported yet
## .
## Numbers are specified in bytes.
## The size of the capacity list should be the same with the dir size.
## For example:
# capacity = [ 10737418240, 10737418240 ]

[storage.latest]
## The list of directories to store the latest data. About 10% of the
## total data is stored in
## the directory list. The directories (or directory) listed here
## require higher IOPS
## metrics than those in storage.main.dir.
## If it is not set (by default), the values of storage.main.dir are
## used.
# dir = [ ]
## The maximum storage capacity of each directory in storage.latest.dir.
## If it is not set, or is set to multiple 0, the actual disk (the disk
## where the directory is located) capacity is used.
# capacity = [ 10737418240, 10737418240 ]

## [storage.io_rate_limit] settings are new in v5.2.0.
[storage.io_rate_limit]
## This configuration item determines whether to limit the I/O traffic,
## which is disabled by default. This traffic limit in TiFlash is
## suitable for cloud storage that has the disk bandwidth of a small
## and specific size.
## The total I/O bandwidth for disk reads and writes. The unit is bytes
## and the default value is 0, which means the I/O traffic is not
## limited by default.
# max_bytes_per_sec = 0
## max_read_bytes_per_sec and max_write_bytes_per_sec have similar
## meanings to max_bytes_per_sec. max_read_bytes_per_sec means the
## total I/O bandwidth for disk reads, and max_write_bytes_per_sec
## means the total I/O bandwidth for disk writes.

```

```

## These configuration items limit I/O bandwidth for disk reads and
→ writes separately. You can use them for cloud storage that
→ calculates the limit of I/O bandwidth for disk reads and writes
→ separately, such as the Persistent Disk provided by Google Cloud
→ Platform.
## When the value of max_bytes_per_sec is not 0, max_bytes_per_sec is
→ prioritized.
# max_read_bytes_per_sec = 0
# max_write_bytes_per_sec = 0

## The following parameters control the bandwidth weights assigned to
→ different I/O traffic types. Generally, you do not need to adjust
→ these parameters.
## TiFlash internally divides I/O requests into four types: foreground
→ writes, background writes, foreground reads, background reads.
## When the I/O traffic limit is initialized, TiFlash assigns the
→ bandwidth according to the following weight ratio.
## The following default configurations indicate that each type of
→ traffic gets a weight of 25% (25 / (25 + 25 + 25 + 25) = 25%).
## If the weight is configured to 0, the corresponding I/O traffic is
→ not limited.
# foreground_write_weight = 25
# background_write_weight = 25
# foreground_read_weight = 25
# background_read_weight = 25
## TiFlash supports automatically tuning the traffic limit for different
→ I/O types according to the current I/O load. Sometimes, the tuned
→ bandwidth might exceed the weight ratio set above.
## auto_tune_sec indicates the interval of automatic tuning. The unit is
→ seconds. If the value of auto_tune_sec is 0, the automatic tuning
→ is disabled.
# auto_tune_sec = 5

[flash]
tidb_status_addr = TiDB status port and address. # Multiple addresses
→ are separated with commas.
service_addr = The listening address of TiFlash Raft services and
→ coprocessor services.

##### Multiple TiFlash nodes elect a master to add or delete placement rules
→ to PD,
##### and the configurations in flash.flash_cluster control this process.
[flash.flash_cluster]
refresh_interval = Master regularly refreshes the valid period.
update_rule_interval = Master regularly gets the status of TiFlash

```

```

    ↪ replicas and interacts with PD.
master_ttl = The valid period of the elected master.
cluster_manager_path = The absolute path of the pd buddy directory.
log = The pd buddy log path.

[flash.proxy]
addr = The listening address of proxy.
advertise-addr = The external access address of addr. If it is left
    ↪ empty, addr is used by default.
data-dir = The data storage path of proxy.
config = The proxy configuration file path.
log-file = The proxy log path.
log-level = The proxy log level. "info" is used by default.
status-addr = The listening address from which the proxy metrics |
    ↪ status information is pulled.
advertise-status-addr = The external access address of status-addr. If
    ↪ it is left empty, status-addr is used by default.

[logger]
level = log level (available options: trace, debug, information, warning
    ↪ , error).
log = The TiFlash log path.
errorlog = The TiFlash error log path.
size = The size of a single log file.
count = The maximum number of log files to save.

[raft]
## PD service address. Multiple addresses are separated with commas.
pd_addr = "10.0.1.11:2379,10.0.1.12:2379,10.0.1.13:2379"

[status]
metrics_port = The port through which Prometheus pulls metrics
    ↪ information.

[profiles]

[profiles.default]
## The default value is true. This parameter determines whether the
    ↪ segment
## of DeltaTree Storage Engine uses logical split.
## Using the logical split can reduce the write amplification, and
    ↪ improve the write speed.
## However, these are at the cost of disk space waste.
dt_enable_logical_split = true

```

```

## The memory usage limit for the generated intermediate data when a
    ↪ single
## coprocessor query is executed. The default value is 0, which means no
    ↪ limit.
max_memory_usage = 0

## The memory usage limit for the generated intermediate data when all
    ↪ queries
## are executed. The default value is 0 (in bytes), which means no limit
    ↪ .
max_memory_usage_for_all_queries = 0

## New in v5.0. This item specifies the maximum number of cop requests
    ↪ that TiFlash Coprocessor executes at the same time. If the number
    ↪ of requests exceeds the specified value, the exceeded requests
    ↪ will queue. If the configuration value is set to 0 or not set, the
    ↪ default value is used, which is twice the number of physical
    ↪ cores.
cop_pool_size = 0
## New in v5.0. This item specifies the maximum number of batch requests
    ↪ that TiFlash Coprocessor executes at the same time. If the number
    ↪ of requests exceeds the specified value, the exceeded requests
    ↪ will queue. If the configuration value is set to 0 or not set, the
    ↪ default value is used, which is twice the number of physical
    ↪ cores.
batch_cop_pool_size = 0

#### Security settings take effect starting from v4.0.5.
[security]
    ## New in v5.0. This configuration item enables or disables log
        ↪ redaction. If the configuration value
    ## is set to true, all user data in the log will be replaced by ?.
    ## Note that you also need to set security.redact-info-log for tiflash-
        ↪ learner's logging in tiflash-learner.toml.
# redact_info_log = false

    ## Path of the file that contains a list of trusted SSL CAs. If set, the
        ↪ following settings
    ## cert_path and key_path are also needed.
# ca_path = "/path/to/ca.pem"
    ## Path of the file that contains X509 certificate in PEM format.
# cert_path = "/path/to/tiflash-server.pem"
    ## Path of the file that contains X509 key in PEM format.
# key_path = "/path/to/tiflash-server-key.pem"

```

```
## New in v5.0. This configuration item enables or disables log
    ↪ redaction. If the configuration value
## is set to true, all user data in the log will be replaced by ?.
## Note that you also need to set security.redact-info-log for tiflash-
    ↪ learner's logging in tiflash-learner.toml.
# redact_info_log = false
```

11.9.3.2.2 Configure the tiflash-learner.toml file

```
[server]
  engine-addr = The external access address of the TiFlash coprocessor
    ↪ service.

[raftstore]
  ## Specifies the number of threads that handle snapshots.
  ## The default number is 2.
  ## If you set it to 0, the multi-thread optimization is disabled.
  snap-handle-pool-size = 2

  ## Specifies the shortest interval at which Raft store persists WAL.
  ## You can properly increase the latency to reduce IOPS usage.
  ## The default value is "4ms".
  ## If you set it to 0ms, the optimization is disabled.
  store-batch-retry-recv-timeout = "4ms"

[security]
  ## New in v5.0. This configuration item enables or disables log
    ↪ redaction.
  ## If the configuration value is set to true,
  ## all user data in the log will be replaced by ?. The default value is
    ↪ false.
  redact-info-log = false
```

In addition to the items above, other parameters are the same with those of TiKV. Note that the configuration items in `tiflash.toml` [`flash.proxy`] will override the overlapping parameters in `tiflash-learner.toml`; The label whose key is `engine` is reserved and cannot be configured manually.

11.9.3.2.3 Multi-disk deployment

TiFlash supports multi-disk deployment. If there are multiple disks in your TiFlash node, you can make full use of those disks by configuring the parameters described in the following sections. For TiFlash's configuration template to be used for TiUP, see [The complex template for the TiFlash topology](#).

Multi-disk deployment with TiDB version earlier than v4.0.9

For TiDB clusters earlier than v4.0.9, TiFlash only supports storing the main data of the storage engine on multiple disks. You can set up a TiFlash node on multiple disks by specifying the `path` (`data_dir` in TiUP) and `path_realtime_mode` configuration.

If there are multiple data storage directories in `path`, separate each with a comma. For example, `/nvme_ssd_a/data/tiflash,/sata_ssd_b/data/tiflash,/sata_ssd_c/data/` → `tiflash`. If there are multiple disks in your environment, it is recommended that each directory corresponds to one disk and you put disks with the best performance at the front to maximize the performance of all disks.

If there are multiple disks with similar I/O metrics on your TiFlash node, you can leave the `path_realtime_mode` parameter to the default value (or you can explicitly set it to `false` →). It means that data will be evenly distributed among all storage directories. However, the latest data is written only to the first directory, so the corresponding disk is busier than other disks.

If there are multiple disks with different I/O metrics on your TiFlash node, it is recommended to set `path_realtime_mode` to `true` and put disks with the best I/O metrics at the front of `path`. It means that the first directory only stores the latest data, and the older data are evenly distributed among the other directories. Note that in this case, the capacity of the first directory should be planned as 10% of the total capacity of all directories.

Multi-disk deployment with TiDB v4.0.9 or later

For TiDB clusters with v4.0.9 or later versions, TiFlash supports storing the main data and the latest data of the storage engine on multiple disks. If you want to deploy a TiFlash node on multiple disks, it is recommended to specify your storage directories in the `[storage]` section to make full use of your node. Note that the configurations earlier than v4.0.9 (`path` and `path_realtime_mode`) are still supported.

If there are multiple disks with similar I/O metrics on your TiFlash node, it is recommended to specify corresponding directories in the `storage.main.dir` list and leave `storage.latest.dir` empty. TiFlash will distribute I/O pressure and data among all directories.

If there are multiple disks with different I/O metrics on your TiFlash node, it is recommended to specify directories with higher metrics in the `storage.latest.dir` list, and specify directories with lower metrics in the `storage.main.dir` list. For example, for one NVMe-SSD and two SATA-SSDs, you can set `storage.latest.dir` to `["/nvme_ssd_a/data/tiflash"]` and `storage.main.dir` to `["/sata_ssd_b/data/tiflash", "/sata_ssd_c/data/tiflash"]`. TiFlash will distribute I/O pressure and data among these two directories list respectively. Note that in this case, the capacity of `storage.latest.dir` should be planned as 10% of the total planned capacity.

Warning:

- The `[storage]` configuration is supported in TiUP since v1.2.5. If your TiDB cluster version is v4.0.9 or later, make sure that your TiUP version

is v1.2.5 or later. Otherwise, the data directories defined in [storage] will not be managed by TiUP.

- After using the `storage` configurations, downgrading your cluster to a version earlier than v4.0.9 might cause **data loss** on TiFlash..

11.9.4 PD Configuration File

The PD configuration file supports more options than command-line parameters. You can find the default configuration file [here](#).

This document only describes parameters that are not included in command-line parameters. Check [here](#) for the command line parameters.

11.9.4.0.1 `name`

- The unique name of a PD node
- Default value: "pd"
- To start multiply PD nodes, use a unique name for each node.

11.9.4.0.2 `data-dir`

- The directory in which PD stores data
- Default value: `default.${name}"`

11.9.4.0.3 `client-urls`

- The list of client URLs to be listened to by PD
- Default value: "`http://127.0.0.1:2379`"
- When you deploy a cluster, you must specify the IP address of the current host as `client-urls` (for example, "`http://192.168.100.113:2379`"). If the cluster runs on Docker, specify the IP address of Docker as "`http://0.0.0.0:2379`".

11.9.4.0.4 `advertise-client-urls`

- The list of advertise URLs for the client to access PD
- Default value: " `${client-urls}`"
- In some situations such as in the Docker or NAT network environment, if a client cannot access PD through the default client URLs listened to by PD, you must manually set the advertise client URLs.
- For example, the internal IP address of Docker is `172.17.0.1`, while the IP address of the host is `192.168.100.113` and the port mapping is set to `-p 2380:2380`. In this case, you can set `advertise-client-urls` to "`http://192.168.100.113:2380`". The client can find this service through "`http://192.168.100.113:2380`".

11.9.4.0.5 `peer-urls`

- The list of peer URLs to be listened to by a PD node
- Default value: "http://127.0.0.1:2380"
- When you deploy a cluster, you must specify `peer-urls` as the IP address of the current host, such as "http://192.168.100.113:2380". If the cluster runs on Docker, specify the IP address of Docker as "http://0.0.0.0:2380".

11.9.4.0.6 `advertise-peer-urls`

- The list of advertise URLs for other PD nodes (peers) to access a PD node
- Default: "\${peer-urls}"
- In some situations such as in the Docker or NAT network environment, if the other nodes (peers) cannot access the PD node through the default peer URLs listened to by this PD node, you must manually set the advertise peer URLs.
- For example, the internal IP address of Docker is 172.17.0.1, while the IP address of the host is 192.168.100.113 and the port mapping is set to -p 2380:2380. In this case, you can set `advertise-peer-urls` to "http://192.168.100.113:2380". The other PD nodes can find this service through "http://192.168.100.113:2380".

11.9.4.0.7 `initial-cluster`

- The initial cluster configuration for bootstrapping
- Default value: "{name}=http://{advertise-peer-url}"
- For example, if name is "pd", and `advertise-peer-urls` is "http://192.168.100.113:2380" ↵ , the `initial-cluster` is "pd=http://192.168.100.113:2380".
- If you need to start three PD servers, the `initial-cluster` might be:

```
pd1=http://192.168.100.113:2380, pd2=http://192.168.100.114:2380, pd3
    ↵ =192.168.100.115:2380
```

11.9.4.0.8 `initial-cluster-state`

- The initial state of the cluster
- Default value: "new"

11.9.4.0.9 `initial-cluster-token`

- Identifies different clusters during the bootstrap phase.
- Default value: "pd-cluster"
- If multiple clusters that have nodes with same configurations are deployed successively, you must specify different tokens to isolate different cluster nodes.

11.9.4.0.10 `lease`

- The timeout of the PD Leader Key lease. After the timeout, the system re-elects a Leader.
- Default value: 3
- Unit: second

11.9.4.0.11 `quota-backend-bytes`

- The storage size of the meta-information database, which is 8GiB by default
- Default value: 8589934592

11.9.4.0.12 `auto-compaction-mod`

- The automatic compaction modes of the meta-information database
- Available options: `periodic` (by cycle) and `revision` (by version number).
- Default value: `periodic`

11.9.4.0.13 `auto-compaction-retention`

- The time interval for automatic compaction of the meta-information database when `auto-compaction-retention` is `periodic`. When the compaction mode is set to `revision`, this parameter indicates the version number for the automatic compaction.
- Default value: 1h

11.9.4.0.14 `force-new-cluster`

- Determines whether to force PD to start as a new cluster and modify the number of Raft members to 1
- Default value: `false`

11.9.4.1 `security`

Configuration items related to security

11.9.4.1.1 `cacert-path`

- The path of the CA file
- Default value: “”

11.9.4.1.2 cert-path

- The path of the Privacy Enhanced Mail (PEM) file that contains the X509 certificate
- Default value: “”

11.9.4.1.3 key-path

- The path of the PEM file that contains the X509 key
- Default value: “”

11.9.4.1.4 redact-info-log New in v5.0

- Controls whether to enable log redaction in the PD log.
- When you set the configuration value to `true`, user data is redacted in the PD log.
- Default value: `false`

11.9.4.2 log

Configuration items related to log

11.9.4.2.1 level

- The log level, which can be specified as “DEBUG”, “INFO”, “WARNING”, “ERROR”, “CRITICAL”.
- Default value: “INFO”

11.9.4.2.2 format

- The log format, which can be specified as “text”, “json”, or “console”
- Default value: “text”

11.9.4.2.3 disable-timestamp

- Whether to disable the automatically generated timestamp in the log
- Default value: `false`

11.9.4.3 log.file

Configuration items related to the log file

11.9.4.3.1 max-size

- The maximum size of a single log file. When this value is exceeded, the system automatically splits the log into several files.
- Default value: 300
- Unit: MiB
- Minimum value: 1

11.9.4.3.2 max-days

- The maximum number of days in which a log is kept
- Default value: 0

11.9.4.3.3 max-backups

- The maximum number of log files to keep
- Default value: 0

11.9.4.4 metric

Configuration items related to monitoring

11.9.4.4.1 interval

- The interval at which monitoring metric data is pushed to Prometheus
- Default value: 15s

11.9.4.5 schedule

Configuration items related to scheduling

11.9.4.5.1 max-merge-region-size

- Controls the size limit of Region Merge. When the Region size is greater than the specified value, PD does not merge the Region with the adjacent Regions.
- Default value: 20

11.9.4.5.2 max-merge-region-keys

- Specifies the upper limit of the Region Merge key. When the Region key is greater than the specified value, the PD does not merge the Region with its adjacent Regions.
- Default value: 200000

11.9.4.5.3 `patrol-region-interval`

- Controls the running frequency at which `replicaChecker` checks the health state of a Region. The smaller this value is, the faster `replicaChecker` runs. Normally, you do not need to adjust this parameter.
- Default value: 10ms

11.9.4.5.4 `split-merge-interval`

- Controls the time interval between the `split` and `merge` operations on the same Region. That means a newly split Region will not be merged for a while.
- Default value: 1h

11.9.4.5.5 `max-snapshot-count`

- Controls the maximum number of snapshots that a single store receives or sends at the same time. PD schedulers depend on this configuration to prevent the resources used for normal traffic from being preempted.
- Default value: 64

11.9.4.5.6 `max-pending-peer-count`

- Controls the maximum number of pending peers in a single store. PD schedulers depend on this configuration to prevent too many Regions with outdated logs from being generated on some nodes.
- Default value: 64

11.9.4.5.7 `max-store-down-time`

- The downtime after which PD judges that the disconnected store can not be recovered. When PD fails to receive the heartbeat from a store after the specified period of time, it adds replicas at other nodes.
- Default value: 30m

11.9.4.5.8 `leader-schedule-limit`

- The number of Leader scheduling tasks performed at the same time
- Default value: 4

11.9.4.5.9 `region-schedule-limit`

- The number of Region scheduling tasks performed at the same time
- Default value: 2048

11.9.4.5.10 `hot-region-schedule-limit`

- Controls the hot Region scheduling tasks that are running at the same time. It is independent of the Region scheduling.
- Default value: 4

11.9.4.5.11 `hot-region-cache-hits-threshold`

- The threshold used to set the number of minutes required to identify a hot Region. PD can participate in the hotspot scheduling only after the Region is in the hotspot state for more than this number of minutes.
- Default value: 3

11.9.4.5.12 `replica-schedule-limit`

- The number of Replica scheduling tasks performed at the same time
- Default value: 64

11.9.4.5.13 `merge-schedule-limit`

- The number of the Region Merge scheduling tasks performed at the same time. Set this parameter to 0 to disable Region Merge.
- Default value: 8

11.9.4.5.14 `high-space-ratio`

- The threshold ratio below which the capacity of the store is sufficient. If the space occupancy ratio of the store is smaller than this threshold value, PD ignores the remaining space of the store when performing scheduling, and balances load mainly based on the Region size. This configuration takes effect only when `region-score-formula-
→ version` is set to v1.
- Default value: 0.7
- Minimum value: greater than 0
- Maximum value: less than 1

11.9.4.5.15 `low-space-ratio`

- The threshold ratio above which the capacity of the store is insufficient. If the space occupancy ratio of a store exceeds this threshold value, PD avoids migrating data to this store as much as possible. Meanwhile, to avoid the disk space of the corresponding store being exhausted, PD performs scheduling mainly based on the remaining space of the store.
- Default value: 0.8
- Minimum value: greater than 0
- Maximum value: less than 1

11.9.4.5.16 `tolerant-size-ratio`

- Controls the `balance` buffer size
- Default value: 0 (automatically adjusts the buffer size)
- Minimum value: 0

11.9.4.5.17 `enable-cross-table-merge`

- Determines whether to enable the merging of cross-table Regions
- Default value: `true`

11.9.4.5.18 `region-score-formula-version` New in v5.0

- Controls the version of the Region score formula
- Default value: `v2`
- Optional values: `v1` and `v2`. Compared to `v1`, the changes in `v2` are smoother, and the scheduling jitter caused by space reclaim is improved.

Note:

If you have upgraded your cluster from a TiDB 4.0 version to the current version, the new formula version is automatically disabled by default to ensure consistent PD behavior before and after the upgrading. If you want to change the formula version, you need to manually switch through the `pd-ctl` setting. For details, refer to [PD Control](#).

11.9.4.5.19 `enable-joint-consensus` New in v5.0

- Controls whether to use Joint Consensus for replica scheduling. If this configuration is disabled, PD schedules one replica at a time.
- Default value: `true`

11.9.4.6 `replication`

Configuration items related to replicas

11.9.4.6.1 `max-replicas`

- The number of replicas, that is, the sum of the number of leaders and followers. The default value 3 means 1 leader and 2 followers. When this configuration is modified online, PD will schedule Regions in the background so that the number of replicas matches this configuration.
- Default value: 3

11.9.4.6.2 `location-labels`

- The topology information of a TiKV cluster
- Default value: `[]`
- [Cluster topology configuration](#)

11.9.4.6.3 `isolation-level`

- The minimum topological isolation level of a TiKV cluster
- Default value: `""`
- [Cluster topology configuration](#)

11.9.4.6.4 `strictly-match-label`

- Enables the strict check for whether the TiKV label matches PD's `location-labels`.
- Default value: `false`

11.9.4.6.5 `enable-placement-rules`

- Enables `placement-rules`.
- Default value: `false`
- See [Placement Rules](#).
- An experimental feature of TiDB 4.0.

11.9.4.6.6 `flow-round-by-digit` New in TiDB 5.1

- Default value: 3
- PD rounds the lowest digits of the flow number, which reduces the update of statistics caused by the changes of the Region flow information. This configuration item is used to specify the number of lowest digits to round for the Region flow information. For example, the flow 100512 will be rounded to 101000 because the default value is 3. This configuration replaces `trace-region-flow`.

Note:

If you have upgraded your cluster from a TiDB 4.0 version to the current version, the behavior of `flow-round-by-digit` after the upgrading and the behavior of `trace-region-flow` before the upgrading are consistent by default. This means that if the value of `trace-region-flow` is false before the upgrading, the value of `flow-round-by-digit` after the upgrading is 127; if the value of `trace-region-flow` is true before the upgrading, the value of `flow-round-by-digit` after the upgrading is 3.

11.9.4.7 `label-property`

Configuration items related to labels

11.9.4.7.1 `key`

- The label key for the store that rejected the Leader
- Default value: ""

11.9.4.7.2 `value`

- The label value for the store that rejected the Leader
- Default value: ""

11.9.4.8 `dashboard`

Configuration items related to the [TiDB Dashboard](#) built in PD.

11.9.4.8.1 `tidb-cacert-path`

- The path of the root CA certificate file. You can configure this path when you connect to TiDB's SQL services using TLS.
- Default value: ""

11.9.4.8.2 `tidb-cert-path`

- The path of the SSL certificate file. You can configure this path when you connect to TiDB's SQL services using TLS.
- Default value: ""

11.9.4.8.3 `tidb-key-path`

- The path of the SSL private key file. You can configure this path when you connect to TiDB's SQL services using TLS.
- Default value: ""

11.9.4.8.4 `public-path-prefix`

- When TiDB Dashboard is accessed behind a reverse proxy, this item sets the public URL path prefix for all web resources.
- Default value: `/dashboard`
- Do **not** modify this configuration item when TiDB Dashboard is accessed not behind a reverse proxy; otherwise, access issues might occur. See [Use TiDB Dashboard behind a Reverse Proxy](#) for details.

11.9.4.8.5 enable-telemetry

- Determines whether to enable the telemetry collection feature in TiDB Dashboard.
- Default value: `true`
- See [Telemetry](#) for details.

11.9.4.9 replication-mode

Configuration items related to the replication mode of all Regions. See [Enable the DR Auto-Sync mode](#) for details.

11.10 System Variables

TiDB system variables behave similar to MySQL with some differences, in that settings might apply on a SESSION, INSTANCE, or GLOBAL scope, or on a scope that combines SESSION, INSTANCE, or GLOBAL.

- Changes to GLOBAL scoped variables **only apply to new connection sessions with TiDB**. Currently active connection sessions are not affected. These changes are persisted and valid after restarts.
- Changes to INSTANCE scoped variables apply to all active or new connection sessions with the current TiDB instance immediately after the changes are made. Other TiDB instances are not affected. These changes are not persisted and become invalid after TiDB restarts.
- Variables can also have NONE scope. These variables are read-only, and are typically used to convey static information that will not change after a TiDB server has started.

Variables can be set with the [SET statement](#) on a per-session, instance or global basis:

```
## These two identical statements change a session variable
SET tidb_distsql_scan_concurrency = 10;
SET SESSION tidb_distsql_scan_concurrency = 10;

## These two identical statements change a global variable
SET @@global.tidb_distsql_scan_concurrency = 10;
SET GLOBAL tidb_distsql_scan_concurrency = 10;
```

Note:

Executing `SET GLOBAL` applies immediately on the TiDB server where the statement was issued. A notification is then sent to all TiDB servers to refresh their system variable cache, which will start immediately as a background

operation. Because there is a risk that some TiDB servers might miss the notification, the system variable cache is also refreshed automatically every 30 seconds. This helps ensure that all servers are operating with the same configuration.

TiDB differs from MySQL in that GLOBAL scoped variables **persist** through TiDB server restarts. Additionally, TiDB presents several MySQL variables as both readable and settable. This is required for compatibility, because it is common for both applications and connectors to read MySQL variables. For example, JDBC connectors both read and set query cache settings, despite not relying on the behavior.

Note:

Larger values do not always yield better performance. It is also important to consider the number of concurrent connections that are executing statements, because most settings apply to each connection.

Consider the unit of a variable when you determine safe values:

- For threads, safe values are typically up to the number of CPU cores.
- For bytes, safe values are typically less than the amount of system memory.
- For time, pay attention that the unit might be seconds or milliseconds.

Variables using the same unit might compete for the same set of resources.

11.10.1 Variable Reference

11.10.1.1 allow_auto_random_explicit_insert New in v4.0.3

- Scope: SESSION | GLOBAL
- Default value: OFF
- Determines whether to allow explicitly specifying the values of the column with the AUTO_RANDOM attribute in the INSERT statement.

11.10.1.2 auto_increment_increment

- Scope: SESSION | GLOBAL
- Default value: 1

- Range: [1, 65535]
- Controls the step size of AUTO_INCREMENT values to be allocated to a column. It is often used in combination with auto_increment_offset.

11.10.1.3 auto_increment_offset

- Scope: SESSION | GLOBAL
- Default value: 1
- Range: [1, 65535]
- Controls the initial offset of AUTO_INCREMENT values to be allocated to a column. This setting is often used in combination with auto_increment_increment. For example:

```
mysql> CREATE TABLE t1 (a int not null primary key auto_increment);
Query OK, 0 rows affected (0.10 sec)

mysql> set auto_increment_offset=1;
Query OK, 0 rows affected (0.00 sec)

mysql> set auto_increment_increment=3;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO t1 VALUES (),(),(),();
Query OK, 4 rows affected (0.04 sec)
Records: 4 Duplicates: 0 Warnings: 0

mysql> SELECT * FROM t1;
+---+
| a |
+---+
| 1 |
| 4 |
| 7 |
| 10 |
+---+
4 rows in set (0.00 sec)
```

11.10.1.4 autocommit

- Scope: SESSION | GLOBAL
- Default value: ON
- Controls whether statements should automatically commit when not in an explicit transaction. See [Transaction Overview](#) for more information.

11.10.1.5 block_encryption_mode

- Scope: SESSION | GLOBAL
- Default value: aes-128-ecb
- Defines the encryption mode for the `AES_ENCRYPT()` and `AES_DECRYPT()` functions.

11.10.1.6 character_set_client

- Scope: SESSION | GLOBAL
- Default value: utf8mb4
- The character set for data sent from the client. See [Character Set and Collation](#) for details on the use of character sets and collations in TiDB. It is recommended to use `SET NAMES` to change the character set when needed.

11.10.1.7 character_set_connection

- Scope: SESSION | GLOBAL
- Default value: utf8mb4
- The character set for string literals that do not have a specified character set.

11.10.1.8 character_set_database

- Scope: SESSION | GLOBAL
- Default value: utf8mb4
- This variable indicates the character set of the default database in use. **It is NOT recommended to set this variable.** When a new default database is selected, the server changes the variable value.

11.10.1.9 character_set_results

- Scope: SESSION | GLOBAL
- Default value: utf8mb4
- The character set that is used when data is sent to the client.

11.10.1.10 character_set_server

- Scope: SESSION | GLOBAL
- Default value: utf8mb4
- The default character set for the server.

11.10.1.11 `collation_connection`

- Scope: SESSION | GLOBAL
- Default value: `utf8mb4_bin`
- This variable indicates the collation for string literals that do not have a specified collation.

11.10.1.12 `collation_database`

- Scope: SESSION | GLOBAL
- Default value: `utf8mb4_bin`
- This variable indicates the collation of the default database in use. **It is NOT recommended to set this variable.** When a new default database is selected, the server changes the variable value.

11.10.1.13 `collation_server`

- Scope: SESSION | GLOBAL
- Default value: `utf8mb4_bin`
- The default collation for the server.

11.10.1.14 `cte_max_recursion_depth`

- Scope: SESSION | GLOBAL
- Default value: 1000
- Range: [0, 4294967295]
- Controls the maximum recursion depth in Common Table Expressions.

11.10.1.15 `datadir`

- Scope: NONE
- Default value: `/tmp/tidb`
- This variable indicates the location where data is stored. This location can be a local path or point to a PD server if the data is stored on TiKV.
- A value in the format of `ip_address:port` indicates the PD server that TiDB connects to on startup.

11.10.1.16 `ddl_slow_threshold`

- Scope: INSTANCE
- Default value: 300
- Unit: Milliseconds
- Log DDL operations whose execution time exceeds the threshold value.

11.10.1.17 default_authentication_plugin

- Scope: GLOBAL
- Default value: `mysql_native_password`
- Possible values: `mysql_native_password`, `caching_sha2_password`
- This variable sets the authentication method that the server advertises when the server-client connection is being established. Possible values for this variable are documented in [Authentication plugin status](#).
- Value options: `mysql_native_password` and `caching_sha2_password`. For more details, see [Authentication plugin status](#).

11.10.1.18 default_week_format

- Scope: SESSION | GLOBAL
- Default value: 0
- Range: [0, 7]
- Sets the week format used by the WEEK() function.

11.10.1.19 foreign_key_checks

- Scope: SESSION | GLOBAL
- Default value: OFF
- For compatibility, TiDB returns foreign key checks as OFF.

11.10.1.20 group_concat_max_len

- Scope: SESSION | GLOBAL
- Default value: 1024
- Range: [4, 18446744073709551615]
- The maximum buffer size for items in the GROUP_CONCAT() function.

11.10.1.21 have_openssl

- Scope: NONE
- Default value: DISABLED
- A read-only variable for MySQL compatibility. Set to YES by the server when the server has TLS enabled.

11.10.1.22 have_ssl

- Scope: NONE
- Default value: DISABLED
- A read-only variable for MySQL compatibility. Set to YES by the server when the server has TLS enabled.

11.10.1.23 `hostname`

- Scope: NONE
- Default value: (system hostname)
- The hostname of the TiDB server as a read-only variable.

11.10.1.24 `identity`

This variable is an alias for `last_insert_id`.

11.10.1.25 `init_connect`

- Scope: GLOBAL
- Default value: “”
- The `init_connect` feature permits a SQL statement to be automatically executed when you first connect to a TiDB server. If you have the CONNECTION_ADMIN or SUPER privileges, this `init_connect` statement will not be executed. If the `init_connect` statement results in an error, your user connection will be terminated.

11.10.1.26 `innodb_lock_wait_timeout`

- Scope: SESSION | GLOBAL
- Default value: 50
- Range: [1, 1073741824]
- Unit: Seconds
- The lock wait timeout for pessimistic transactions (default).

11.10.1.27 `interactive_timeout`

- Scope: SESSION | GLOBAL
- Default value: 28800
- Range: [1, 31536000]
- Unit: Seconds
- This variable represents the idle timeout of the interactive user session. Interactive user session refers to the session established by calling `mysql_real_connect()` API using the `CLIENT_INTERACTIVE` option (for example, MySQL Shell and MySQL Client). This variable is fully compatible with MySQL.

11.10.1.28 `last_insert_id`

- Scope: SESSION
- Default value: 0

- This variable returns the last `AUTO_INCREMENT` or `AUTO_RANDOM` value generated by an insert statement.
- The value of `last_insert_id` is the same as the value returned by the function `LAST_INSERT_ID()`.

11.10.1.29 `last_plan_from_binding` New in v4.0

- Scope: SESSION
- Default value: OFF
- This variable is used to show whether the execution plan used in the previous statement was influenced by a [plan binding](#)

11.10.1.30 `last_plan_from_cache` New in v4.0

- Scope: SESSION
- Default value: OFF
- This variable is used to show whether the execution plan used in the previous `execute` statement is taken directly from the plan cache.

11.10.1.31 `license`

- Scope: NONE
- Default value: Apache License 2.0
- This variable indicates the license of your TiDB server installation.

11.10.1.32 `log_bin`

- Scope: NONE
- Default value: OFF
- This variable indicates whether [TiDB Binlog](#) is used.

11.10.1.33 `max_allowed_packet`

- Scope: SESSION | GLOBAL
- Default value: 67108864
- Range: [1024, 1073741824]
- Unit: Bytes
- The maximum size of a packet for the MySQL protocol.

11.10.1.34 max_execution_time

- Scope: SESSION | GLOBAL
- Default value: 0
- Range: [0, 2147483647]
- Unit: Milliseconds
- The maximum execution time of a statement. The default value is unlimited (zero).

Note:

Unlike in MySQL, the `max_execution_time` system variable currently works on all kinds of statements in TiDB, not only restricted to the `SELECT` statement. The precision of the timeout value is roughly 100ms. This means the statement might not be terminated in accurate milliseconds as you specify.

11.10.1.35 placement_checks

- Scope: SESSION | GLOBAL
- Default value: ON
- This variable controls whether DDL statements validate [Placement Rules in SQL](#).
- It is intended to be used by logical dump/restore tools to ensure that tables can always be created even if placement rules are violated. This is similar to how mysqldump writes `SET FOREIGN_KEY_CHECKS=0;` to the start of every dump file.

11.10.1.36 plugin_dir

- Scope: INSTANCE
- Default value: “”
- Indicates the directory to load plugins as specified by a command-line flag.

11.10.1.37 plugin_load

- Scope: INSTANCE
- Default value: “”
- Indicates the plugins to load when TiDB is started. These plugins are specified by a command-line flag and separated by commas.

11.10.1.38 port

- Scope: NONE
- Default value: 4000
- Range: [0, 65535]
- The port that the `tidb-server` is listening on when speaking the MySQL protocol.

11.10.1.39 skip_name_resolve New in v5.2.0

- Scope: GLOBAL
- Default value: OFF
- This variable controls whether the `tidb-server` instance resolves hostnames as a part of the connection handshake.
- When the DNS is unreliable, you can enable this option to improve network performance.

Note:

When `skip_name_resolve=ON`, users with a hostname in their identity will no longer be able to log into the server. For example:

```
CREATE USER 'appuser'@'apphost' IDENTIFIED BY 'app-password';
```

In this example, it is recommended to replace `apphost` with an IP address or the wildcard (%).

11.10.1.40 socket

- Scope: NONE
- Default value: “”
- The local unix socket file that the `tidb-server` is listening on when speaking the MySQL protocol.

11.10.1.41 sql_log_bin

- Scope: SESSION | GLOBAL
- Default value: ON
- Indicates whether to write changes to **TiDB Binlog** or not.

Note:

It is not recommended to set `sql_log_bin` as a global variable because the future versions of TiDB might only allow setting this as a session variable.

11.10.1.42 `sql_mode`

- Scope: SESSION | GLOBAL
- Default value: `ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,`
 ↳ `NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION`
- This variable controls a number of MySQL compatibility behaviors. See [SQL Mode](#) for more information.

11.10.1.43 `sql_select_limit` New in v4.0.2

- Scope: SESSION | GLOBAL
- Default value: `18446744073709551615`
- Range: `[0, 18446744073709551615]`
- Unit: Rows
- The maximum number of rows returned by the SELECT statements.

11.10.1.44 `ssl_ca`

- Scope: NONE
- Default value: `""`
- The location of the certificate authority file (if there is one).

11.10.1.45 `ssl_cert`

- Scope: NONE
- Default value: `""`
- The location of the certificate file (if there is a file) that is used for SSL/TLS connections.

11.10.1.46 `ssl_key`

- Scope: NONE
- Default value: `""`
- The location of the private key file (if there is one) that is used for SSL/TLS connections.

11.10.1.47 system_time_zone

- Scope: NONE
- Default value: (system dependent)
- This variable shows the system time zone from when TiDB was first bootstrapped. See also [time_zone](#).

11.10.1.48 tidb_allow_batch_cop New in v4.0

- Scope: SESSION | GLOBAL
- Default value: 1
- Range: [0, 2]
- This variable is used to control how TiDB sends a coprocessor request to TiFlash. It has the following values:
 - 0: Never send requests in batches
 - 1: Aggregation and join requests are sent in batches
 - 2: All coprocessor requests are sent in batches

11.10.1.49 tidb_allow_fallback_to_tikv New in v5.0

- Scope: SESSION | GLOBAL
- Default value: “”
- This variable is used to specify a list of storage engines that might fall back to TiKV. If the execution of a SQL statement fails due to a failure of the specified storage engine in the list, TiDB retries executing this SQL statement with TiKV. This variable can be set to “” or “tiflash”. When this variable is set to “tiflash”, if TiFlash returns a timeout error (error code: ErrTiFlashServerTimeout), TiDB retries executing this SQL statement with TiKV.

11.10.1.50 tidb_allow_function_for_expression_index New in v5.2.0

- Scope: NONE
- This variable is used to show the functions that are allowed to be used for creating expression indexes.

11.10.1.51 `tidb_allow_mpp` New in v5.0

- Scope: SESSION | GLOBAL
- Default value: OFF
- Controls whether to use the MPP mode of TiFlash to execute queries. The value options are as follows:
 - 0 or OFF, which means that the MPP mode will not be used.
 - 1 or ON, which means that the optimizer determines whether to use the MPP mode based on the cost estimation (by default).

MPP is a distributed computing framework provided by the TiFlash engine, which allows data exchange between nodes and provides high-performance, high-throughput SQL algorithms. For details about the selection of the MPP mode, refer to [Control whether to select the MPP mode](#).

11.10.1.52 `tidb_allow_remove_auto_inc` New in v2.1.18 and v3.0.4

- Scope: SESSION
- Default value: OFF
- This variable is used to set whether the AUTO_INCREMENT property of a column is allowed to be removed by executing ALTER TABLE MODIFY or ALTER TABLE CHANGE statements. It is not allowed by default.

11.10.1.53 `tidb_analyze_version` New in v5.1.0

- Scope: SESSION | GLOBAL
- Default value: 2
- Range: [1, 2]
- Controls how TiDB collects statistics.
- In versions before v5.1.0, the default value of this variable is 1. In v5.1.0, the default value of this variable is 2, which serves as an experimental feature. For detailed introduction, see [Introduction to Statistics](#).

11.10.1.54 `tidb_auto_analyze_end_time`

- Scope: GLOBAL
- Default value: 23:59 +0000
- This variable is used to restrict the time window that the automatic update of statistics is permitted. For example, to only allow automatic statistics updates between 1AM and 3AM, set `tidb_auto_analyze_start_time='01:00 +0000'` and `tidb_auto_analyze_end_time='03:00 +0000'`.

11.10.1.55 tidb_auto_analyze_ratio

- Scope: GLOBAL
- Default value: 0.5
- This variable is used to set the threshold when TiDB automatically executes `ANALYZE TABLE` in a background thread to update table statistics. For example, a value of 0.5 means that auto-analyze is triggered when greater than 50% of the rows in a table have been modified. Auto-analyze can be restricted to only execute during certain hours of the day by specifying `tidb_auto_analyze_start_time` and `tidb_auto_analyze_end_time`.

Note:

Only when the `run-auto-analyze` option is enabled in the starting configuration file of TiDB, the `auto_analyze` feature can be triggered.

11.10.1.56 tidb_auto_analyze_start_time

- Scope: GLOBAL
- Default value: 00:00 +0000
- This variable is used to restrict the time window that the automatic update of statistics is permitted. For example, to only allow automatic statistics updates between 1AM and 3AM, set `tidb_auto_analyze_start_time='01:00 +0000'` and `tidb_auto_analyze_end_time='03:00 +0000'`.

11.10.1.57 tidb_backoff_lock_fast

- Scope: SESSION | GLOBAL
- Default value: 100
- Range: [1, 2147483647]
- This variable is used to set the `backoff` time when the read request meets a lock.

11.10.1.58 tidb_backoff_weight

- Scope: SESSION | GLOBAL
- Default value: 2
- Range: [1, 2147483647]

- This variable is used to increase the weight of the maximum time of TiDB backoff, that is, the maximum retry time for sending a retry request when an internal network or other component (TiKV, PD) failure is encountered. This variable can be used to adjust the maximum retry time and the minimum value is 1.

For example, the base timeout for TiDB to take TSO from PD is 15 seconds. When `tidb_backoff_weight = 2`, the maximum timeout for taking TSO is: $base\ time * 2 = 30\ seconds$.

In the case of a poor network environment, appropriately increasing the value of this variable can effectively alleviate error reporting to the application end caused by timeout. If the application end wants to receive the error information more quickly, minimize the value of this variable.

11.10.1.59 `tidb_broadcast_join_threshold_count` New in v5.0

- Scope: SESSION | GLOBAL
- Default value: 10240
- Range: [0, 9223372036854775807]
- Unit: Rows
- If the objects of the join operation belong to a subquery, the optimizer cannot estimate the size of the subquery result set. In this situation, the size is determined by the number of rows in the result set. If the estimated number of rows in the subquery is less than the value of this variable, the Broadcast Hash Join algorithm is used. Otherwise, the Shuffled Hash Join algorithm is used.

11.10.1.60 `tidb_broadcast_join_threshold_size` New in v5.0

- Scope: SESSION | GLOBAL
- Default value: 104857600 (100 MiB)
- Range: [0, 9223372036854775807]
- Unit: Bytes
- If the table size is less than the value of the variable, the Broadcast Hash Join algorithm is used. Otherwise, the Shuffled Hash Join algorithm is used.

11.10.1.61 `tidb_build_stats_concurrency`

- Scope: SESSION | GLOBAL
- Default value: 4
- Unit: Threads
- This variable is used to set the concurrency of executing the ANALYZE statement.
- When the variable is set to a larger value, the execution performance of other queries is affected.

11.10.1.62 `tidb_capture_plan_baselines` New in v4.0

- Scope: SESSION | GLOBAL
- Default value: OFF
- This variable is used to control whether to enable the [baseline capturing](#) feature. This feature depends on the statement summary, so you need to enable the statement summary before you use baseline capturing.
- After this feature is enabled, the historical SQL statements in the statement summary are traversed periodically, and bindings are automatically created for SQL statements that appear at least twice.

11.10.1.63 `tidb_check_mb4_value_in_utf8`

- Scope: INSTANCE
- Default value: ON
- This variable is used to enforce that the `utf8` character set only stores values from the [Basic Multilingual Plane \(BMP\)](#). To store characters outside the BMP, it is recommended to use the `utf8mb4` character set.
- You might need to disable this option when upgrading your cluster from an earlier version of TiDB where the `utf8` checking was more relaxed. For details, see [FAQs After Upgrade](#).

11.10.1.64 `tidb_checksum_table_concurrency`

- Scope: SESSION
- Default value: 4
- Unit: Threads
- This variable is used to set the scan index concurrency of executing the `ADMIN ↔ CHECKSUM TABLE` statement.
- When the variable is set to a larger value, the execution performance of other queries is affected.

11.10.1.65 `tidb_config`

- Scope: SESSION
- Default value: “”
- This variable is read-only. It is used to obtain the configuration information of the current TiDB server.

11.10.1.66 tidb_constraint_check_in_place

- Scope: SESSION | GLOBAL
- Default value: OFF
- This setting only applies to optimistic transactions. When this variable is set to OFF → , checking for duplicate values in UNIQUE indexes is deferred until the transaction commits. This helps improve performance, but might be an unexpected behavior for some applications. See [Constraints](#) for details.

- When set to zero and using optimistic transactions:

```
tidb> create table t (i int key);
tidb> insert into t values (1);
tidb> begin optimistic;
tidb> insert into t values (1);
Query OK, 1 row affected
tidb> commit; -- Check only when a transaction is committed.
ERROR 1062 : Duplicate entry '1' for key 'PRIMARY'
```

- When set to 1 and using optimistic transactions:

```
tidb> set @@tidb_constraint_check_in_place=1;
tidb> begin optimistic;
tidb> insert into t values (1);
ERROR 1062 : Duplicate entry '1' for key 'PRIMARY'
```

Constraint checking is always performed in place for pessimistic transactions (default).

11.10.1.67 tidb_current_ts

- Scope: SESSION
- Default value: 0
- This variable is read-only. It is used to obtain the timestamp of the current transaction.

11.10.1.68 tidb_ddl_error_count_limit

- Scope: GLOBAL
- Default value: 512
- Range: [0, 9223372036854775807]
- This variable is used to set the number of retries when the DDL operation fails. When the number of retries exceeds the parameter value, the wrong DDL operation is canceled.

11.10.1.69 tidb_ddl_reorg_batch_size

- Scope: GLOBAL
- Default value: 256
- Range: [32, 10240]
- Unit: Rows
- This variable is used to set the batch size during the `re-organize` phase of the DDL operation. For example, when TiDB executes the `ADD INDEX` operation, the index data needs to be backfilled by `tidb_ddl_reorg_worker_cnt` (the number) concurrent workers. Each worker backfills the index data in batches.
 - If many updating operations such as `UPDATE` and `REPLACE` exist during the `ADD INDEX` operation, a larger batch size indicates a larger probability of transaction conflicts. In this case, you need to adjust the batch size to a smaller value. The minimum value is 32.
 - If the transaction conflict does not exist, you can set the batch size to a large value. This can increase the speed of the backfilling data, but the write pressure on TiKV also becomes higher.

11.10.1.70 tidb_ddl_reorg_priority

- Scope: SESSION
- Default value: `PRIORITY_LOW`
- This variable is used to set the priority of executing the `ADD INDEX` operation in the `re-organize` phase.
- You can set the value of this variable to `PRIORITY_LOW`, `PRIORITY_NORMAL` or `PRIORITY_HIGH`.

11.10.1.71 tidb_ddl_reorg_worker_cnt

- Scope: GLOBAL
- Default value: 4
- Range: [1, 256]
- Unit: Threads
- This variable is used to set the concurrency of the DDL operation in the `re-organize` phase.

11.10.1.72 tidb_disable_txn_auto_retry

- Scope: SESSION | GLOBAL
- Default value: ON

- This variable is used to set whether to disable the automatic retry of explicit optimistic transactions. The default value of `ON` means that transactions will not automatically retry in TiDB and `COMMIT` statements might return errors that need to be handled in the application layer.

Setting the value to `OFF` means that TiDB will automatically retry transactions, resulting in fewer errors from `COMMIT` statements. Be careful when making this change, because it might result in lost updates.

This variable does not affect automatically committed implicit transactions and internally executed transactions in TiDB. The maximum retry count of these transactions is determined by the value of `tidb_retry_limit`.

For more details, see [limits of retry](#).

This variable only applies to optimistic transactions, not to pessimistic transactions. The number of retries for pessimistic transactions is controlled by `max_retry_count`.

11.10.1.73 `tidb_distsql_scan_concurrency`

- Scope: SESSION | GLOBAL
- Default value: 15
- Range: [1, 256]
- Unit: Threads
- This variable is used to set the concurrency of the `scan` operation.
- Use a bigger value in OLAP scenarios, and a smaller value in OLTP scenarios.
- For OLAP scenarios, the maximum value should not exceed the number of CPU cores of all the TiKV nodes.
- If a table has a lot of partitions, you can reduce the variable value appropriately to avoid TiKV becoming out of memory (OOM).

11.10.1.74 `tidb_dml_batch_size`

- Scope: SESSION | GLOBAL
- Default value: 0
- Range: [0, 2147483647]
- Unit: Rows
- When this value is greater than 0, TiDB will batch commit statements such as `INSERT` or `LOAD DATA` into smaller transactions. This reduces memory usage and helps ensure that the `txn-total-size-limit` is not reached by bulk modifications.
- Only the value 0 provides ACID compliance. Setting this to any other value will break the atomicity and isolation guarantees of TiDB.

11.10.1.75 `tidb_enable_1pc` New in v5.0

- Scope: SESSION | GLOBAL

- Default value: `ON`
- This variable is used to specify whether to enable the one-phase commit feature for transactions that only affect one Region. Compared with the often-used two-phase commit, one-phase commit can greatly reduce the latency of transaction commit and increase the throughput.

Note:

- The default value of `ON` only applies to new clusters. If your cluster was upgraded from an earlier version of TiDB, the value `OFF` will be used instead.
- If you have enabled TiDB Binlog, enabling this variable cannot improve the performance. To improve the performance, it is recommended to use [TiCDC](#) instead.
- Enabling this parameter only means that one-phase commit becomes an optional mode of transaction commit. In fact, the most suitable mode of transaction commit is determined by TiDB.

11.10.1.76 `tidb_enable_alter_placement`

Warning:

Currently, Placement Rules in SQL is an experimental feature. It is not recommended that you use it in production environments.

- Scope: `GLOBAL`
- Default value: `OFF`
- This variable enables or disables [Placement Rules in SQL](#).

11.10.1.77 `tidb_enable_amend_pessimistic_txn` New in v4.0.7

- Scope: `SESSION | GLOBAL`
- Default value: `OFF`
- This variable is used to control whether to enable the `AMEND TRANSACTION` feature. If you enable the `AMEND TRANSACTION` feature in a pessimistic transaction, when concurrent DDL operations and SCHEMA VERSION changes exist on tables associated

with this transaction, TiDB attempts to amend the transaction. TiDB corrects the transaction commit to make the commit consistent with the latest valid SCHEMA VERSION so that the transaction can be successfully committed without getting the **Information schema is changed** error. This feature is effective on the following concurrent DDL operations:

- ADD COLUMN or DROP COLUMN operations.
- MODIFY COLUMN or CHANGE COLUMN operations which increase the length of a field.
- ADD INDEX or DROP INDEX operations in which the index column is created before the transaction is opened.

Note:

Currently, this feature is incompatible with TiDB Binlog in some scenarios and might cause semantic changes on a transaction. For more usage precautions of this feature, refer to [Incompatibility issues about transaction semantic](#) and [Incompatibility issues about TiDB Binlog](#).

11.10.1.78 `tidb_enable_async_commit` New in v5.0

- Scope: SESSION | GLOBAL
- Default value: `ON`
- This variable controls whether to enable the async commit feature for the second phase of the two-phase transaction commit to perform asynchronously in the background. Enabling this feature can reduce the latency of transaction commit.

Note:

- The default value of `ON` only applies to new clusters. if your cluster was upgraded from an earlier version of TiDB, the value `OFF` will be used instead.
- If you have enabled TiDB Binlog, enabling this variable cannot improve the performance. To improve the performance, it is recommended to use [TiCDC](#) instead.
- Enabling this parameter only means that Async Commit becomes an optional mode of transaction commit. In fact, the most suitable mode of transaction commit is determined by TiDB.

11.10.1.79 tidb_enable_auto_increment_in_generated

- Scope: SESSION | GLOBAL
- Default value: OFF
- This variable is used to determine whether to include the AUTO_INCREMENT columns when creating a generated column or an expression index.

11.10.1.80 tidb_enable_cascades_planner

Warning:

Currently, cascades planner is an experimental feature. It is not recommended that you use it in production environments.

- Scope: SESSION | GLOBAL
- Default value: OFF
- This variable is used to control whether to enable the cascades planner.

11.10.1.81 tidb_enable_chunk_rpc New in v4.0

- Scope: SESSION
- Default value: ON
- This variable is used to control whether to enable the Chunk data encoding format in Coprocessor.

11.10.1.82 tidb_enable_clustered_index New in v5.0

- Scope: SESSION | GLOBAL
- Default value: INT_ONLY
- Possible values: OFF, ON, INT_ONLY
- This variable is used to control whether to create the primary key as a **clustered index** by default. “By default” here means that the statement does not explicitly specify the keyword CLUSTERED/NONCLUSTERED. Supported values are OFF, ON, and INT_ONLY:
 - OFF indicates that primary keys are created as non-clustered indexes by default.
 - ON indicates that primary keys are created as clustered indexes by default.
 - INT_ONLY indicates that the behavior is controlled by the configuration item `alter → -primary-key`. If `alter-primary-key` is set to `true`, all primary keys are created as non-clustered indexes by default. If it is set to `false`, only the primary keys which consist of an integer column are created as clustered indexes.

11.10.1.83 tidb_enable_collect_execution_info

- Scope: INSTANCE
- Default value: ON
- This variable controls whether to record the execution information of each operator in the slow query log.

11.10.1.84 tidb_enable_enhanced_security

- Scope: NONE
- Default value: OFF
- This variable indicates whether the TiDB server you are connected to has the Security Enhanced Mode (SEM) enabled. To change its value, you need to modify the value of `enable-sem` in your TiDB server configuration file and restart the TiDB server.
- SEM is inspired by the design of systems such as [Security-Enhanced Linux](#). It reduces the abilities of users with the MySQL SUPER privilege and instead requires RESTRICTED fine-grained privileges to be granted as a replacement. These fine-grained privileges include:
 - RESTRICTED_TABLES_ADMIN: The ability to write data to system tables in the `mysql` schema and to see sensitive columns on `information_schema` tables.
 - RESTRICTED_STATUS_ADMIN: The ability to see sensitive variables in the command `SHOW STATUS`.
 - RESTRICTED_VARIABLES_ADMIN: The ability to see and set sensitive variables in `SHOW [GLOBAL] VARIABLES` and `SET`.
 - RESTRICTED_USER_ADMIN: The ability to prevent other users from making changes or dropping a user account.

11.10.1.85 tidb_enable_fast_analyze

Warning:

Currently, `Fast Analyze` is an experimental feature. It is not recommended that you use it in production environments.

- Scope: SESSION | GLOBAL
- Default value: OFF
- This variable is used to set whether to enable the statistics `Fast Analyze` feature.
- If the statistics `Fast Analyze` feature is enabled, TiDB randomly samples about 10,000 rows of data as statistics. When the data is distributed unevenly or the data size is small, the statistics accuracy is low. This might lead to a non-optimal execution plan, for example, selecting a wrong index. If the execution time of the regular `Analyze` statement is acceptable, it is recommended to disable the `Fast Analyze` feature.

11.10.1.86 `tidb_enable_index_merge` New in v4.0

- Scope: SESSION | GLOBAL
- Default value: OFF
- This variable is used to control whether to enable the index merge feature.

11.10.1.87 `tidb_enable_list_partition` New in v5.0

Warning:

Currently, List partition and List COLUMNS partition are experimental features. It is not recommended that you use it in production environments.

- Scope: SESSION | GLOBAL
- Default value: OFF
- This variable is used to set whether to enable the LIST (COLUMNS) TABLE PARTITION feature.

11.10.1.88 `tidb_enable_noop_functions` New in v4.0

- Scope: SESSION | GLOBAL
- Default value: OFF
- Possible values: OFF, ON, WARN
- By default, TiDB returns an error when you attempt to use the syntax for functionality that is not yet implemented. When the variable value is set to ON, TiDB silently ignores such cases of unavailable functionality, which is helpful if you cannot make changes to the SQL code.
- Enabling noop functions controls the following behaviors:
 - `get_lock` and `release_lock` functions
 - `LOCK IN SHARE MODE` syntax
 - `SQL_CALC_FOUND_ROWS` syntax
 - `START TRANSACTION READ ONLY` and `SET TRANSACTION READ ONLY` syntax
 - The `tx_read_only`, `transaction_read_only`, `offline_mode`, `super_read_only` ↗, `read_only` and `sql_auto_is_null` system variables

Warning:

Only the default value of OFF can be considered safe. Setting `tidb_enable_noop_functions=1` might lead to unexpected behaviors in your

application, because it permits TiDB to ignore certain syntax without providing an error. For example, the syntax `START TRANSACTION READ ONLY` is permitted, but the transaction remains in read-write mode.

11.10.1.89 `tidb_enable_parallel_apply` New in v5.0

- Scope: SESSION | GLOBAL
- Default value: OFF
- This variable controls whether to enable concurrency for the Apply operator. The number of concurrencies is controlled by the `tidb_executor_concurrency` variable. The Apply operator processes correlated subqueries and has no concurrency by default, so the execution speed is slow. Setting this variable value to 1 can increase concurrency and speed up execution. Currently, concurrency for Apply is disabled by default.

11.10.1.90 `tidb_enable_pseudo_for_outdated_stats` New in v5.3.0

- Scope: SESSION | GLOBAL
- Default value: ON
- This variable controls the behavior of the optimizer on using statistics of a table when the statistics are outdated.
- The optimizer determines whether the statistics of a table is outdated in this way: since the last time `ANALYZE` is executed on a table to get the statistics, if 80% of the table rows are modified (the modified row count divided by the total row count), the optimizer determines that the statistics of this table is outdated. You can change this ratio using the `pseudo-estimate-ratio` configuration.
- By default (with the variable value ON), when the statistics of a table is outdated, the optimizer determines that the statistics of the table is no longer reliable except for the total row count. Then, the optimizer uses the pseudo statistics. If you set the variable value to OFF, even if the statistics of a table are outdated, the optimizer still keeps using the statistics.
- If the data on a table is frequently modified without executing `ANALYZE` on this table in time, to keep the execution plan stable, you can set the variable value to OFF.

11.10.1.91 `tidb_enable_rate_limit_action`

- Scope: SESSION | GLOBAL
- Default value: ON
- This variable controls whether to enable the dynamic memory control feature for the operator that reads data. By default, this operator enables the maximum number of threads that `tidb_disql_scan_concurrency` allows to read data. When the memory usage of a single SQL statement exceeds `tidb_mem_quota_query` each time, the operator that reads data stops one thread.

- When the operator that reads data has only one thread left and the memory usage of a single SQL statement continues to exceed `tidb_mem_quota_query`, this SQL statement triggers other memory control behaviors, such as [spilling data to disk](#).

11.10.1.92 `tidb_enable_slow_log`

- Scope: INSTANCE
- Default value: ON
- This variable is used to control whether to enable the slow log feature.

11.10.1.93 `tidb_enable_stmt_summary` New in v3.0.4

- Scope: SESSION | GLOBAL
- Default value: ON
- This variable is used to control whether to enable the statement summary feature. If enabled, SQL execution information like time consumption is recorded to the `information_schema.STATEMENTS_SUMMARY` system table to identify and troubleshoot SQL performance issues.

11.10.1.94 `tidb_enable_strict_double_type_check` New in v5.0

- Scope: SESSION | GLOBAL
- Default value: ON
- This variable is used to control if tables can be created with invalid definitions of type DOUBLE. This setting is intended to provide an upgrade path from earlier versions of TiDB, which were less strict in validating types.
- The default value of ON is compatible with MySQL.

For example, the type `DOUBLE(10)` is now considered invalid because the precision of floating point types is not guaranteed. After changing `tidb_enable_strict_double_type_check` to OFF, the table is created:

```
mysql> CREATE TABLE t1 (id int, c double(10));
ERROR 1149 (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

mysql> SET tidb_enable_strict_double_type_check = 'OFF';
Query OK, 0 rows affected (0.00 sec)

mysql> CREATE TABLE t1 (id int, c double(10));
Query OK, 0 rows affected (0.09 sec)
```

Note:

This setting only applies to the type `DOUBLE` since MySQL permits precision for `FLOAT` types. This behavior is deprecated starting with MySQL 8.0.17, and it is not recommended to specify precision for either `FLOAT` or `DOUBLE` types.

11.10.1.95 `tidb_enable_table_partition`

- Scope: SESSION | GLOBAL
- Default value: ON
- Possible values: OFF, ON, AUTO
- This variable is used to set whether to enable the TABLE PARTITION feature:
 - ON indicates enabling Range partitioning, Hash partitioning, and Range column partitioning with one single column.
 - AUTO functions the same way as ON does.
 - OFF indicates disabling the TABLE PARTITION feature. In this case, the syntax that creates a partition table can be executed, but the table created is not a partitioned one.

11.10.1.96 `tidb_enable_telemetry` New in v4.0.2

- Scope: GLOBAL
- Default value: ON
- This variable is used to dynamically control whether the telemetry collection in TiDB is enabled. By setting the value to OFF, the telemetry collection is disabled. If the `enable-telemetry` TiDB configuration item is set to false on all TiDB instances, the telemetry collection is always disabled and this system variable will not take effect. See [Telemetry](#) for details.

11.10.1.97 `tidb_enable_tso_follower_proxy` New in v5.3

- Scope: GLOBAL
- Default value: OFF
- This variable is used to enable the TSO Follower Proxy feature. When the value is OFF, TiDB only gets TSO from the PD leader. After this feature is enabled, TiDB gets TSO by evenly sending requests to all PD nodes and forwarding TSO requests through PD followers. This helps reduce the CPU pressure of PD leader.
- Scenarios for enabling TSO Follower Proxy:

- Due to the high pressure of TSO requests, the CPU of the PD leader reaches a bottleneck, which causes high latency of TSO RPC requests.
- The TiDB cluster has many TiDB instances, and increasing the value of `tidb_tso_client_batch_max_wait_time` cannot alleviate the high latency issue of TSO RPC requests.

Note:

Suppose that the TSO RPC latency increases for reasons other than a CPU usage bottleneck of the PD leader (such as network issues). In this case, enabling the TSO Follower Proxy might increase the execution latency in TiDB and affect the QPS performance of the cluster.

11.10.1.98 `tidb_enable_vectorized_expression` New in v4.0

- Scope: SESSION | GLOBAL
- Default value: ON
- This variable is used to control whether to enable vectorized execution.

11.10.1.99 `tidb_enable_window_function`

- Scope: SESSION | GLOBAL
- Default value: ON
- This variable is used to control whether to enable the support for window functions. Note that window functions may use reserved keywords. This might cause SQL statements that could be executed normally cannot be parsed after upgrading TiDB. In this case, you can set `tidb_enable_window_function` to OFF.

11.10.1.100 `tidb_enforce_mpp` New in v5.1

- Scope: SESSION
- Default value: OFF
- To change this default value, modify the `performance.enforce-mpp` configuration value.
- Controls whether to ignore the optimizer's cost estimation and to forcibly use TiFlash's MPP mode for query execution. The value options are as follows:
 - 0 or OFF, which means that the MPP mode is not forcibly used (by default).
 - 1 or ON, which means that the cost estimation is ignored and the MPP mode is forcibly used. Note that this setting only takes effect when `tidb_allow_mpp=→ true`.

MPP is a distributed computing framework provided by the TiFlash engine, which allows data exchange between nodes and provides high-performance, high-throughput SQL algorithms. For details about the selection of the MPP mode, refer to [Control whether to select the MPP mode](#).

11.10.1.101 `tidb_evolve_plan_baselines` New in v4.0

- Scope: SESSION | GLOBAL
- Default value: OFF
- This variable is used to control whether to enable the baseline evolution feature. For detailed introduction or usage , see [Baseline Evolution](#).
- To reduce the impact of baseline evolution on the cluster, use the following configurations:
 - Set `tidb_evolve_plan_task_max_time` to limit the maximum execution time of each execution plan. The default value is 600s.
 - Set `tidb_evolve_plan_task_start_time` and `tidb_evolve_plan_task_end_time` ↗ to limit the time window. The default values are respectively 00:00 +0000 and 23:59 +0000.

11.10.1.102 `tidb_evolve_plan_task_end_time` New in v4.0

- Scope: GLOBAL
- Default value: 23:59 +0000
- This variable is used to set the end time of baseline evolution in a day.

11.10.1.103 `tidb_evolve_plan_task_max_time` New in v4.0

- Scope: GLOBAL
- Default value: 600
- Range: [-1, 9223372036854775807]
- Unit: Seconds
- This variable is used to limit the maximum execution time of each execution plan in the baseline evolution feature.

11.10.1.104 `tidb_evolve_plan_task_start_time` New in v4.0

- Scope: GLOBAL
- Default value: 00:00 +0000
- This variable is used to set the start time of baseline evolution in a day.

11.10.1.105 `tidb_executor_concurrency` New in v5.0

- Scope: SESSION | GLOBAL
- Default value: 5
- Range: [1, 256]
- Unit: Threads

This variable is used to set the concurrency of the following SQL operators (to one value):

- index lookup
- index lookup join
- hash join
- hash aggregation (the partial and final phases)
- window
- projection

`tidb_executor_concurrency` incorporates the following existing system variables as a whole for easier management:

- `tidb_index_lookup_concurrency`
- `tidb_index_lookup_join_concurrency`
- `tidb_hash_join_concurrency`
- `tidb_hashagg_partial_concurrency`
- `tidb_hashagg_final_concurrency`
- `tidb_projection_concurrency`
- `tidb_window_concurrency`

Since v5.0, you can still separately modify the system variables listed above (with a deprecation warning returned) and your modification only affects the corresponding single operators. After that, if you use `tidb_executor_concurrency` to modify the operator concurrency, the separately modified operators will not be affected. If you want to use `tidb_executor_concurrency` to modify the concurrency of all operators, you can set the values of all variables listed above to -1.

For a system upgraded to v5.0 from an earlier version, if you have not modified any value of the variables listed above (which means that the `tidb_hash_join_concurrency` value is 5 and the values of the rest are 4), the operator concurrency previously managed by these variables will automatically be managed by `tidb_executor_concurrency`. If you have modified any of these variables, the concurrency of the corresponding operators will still be controlled by the modified variables.

11.10.1.106 tidb_expensive_query_time_threshold

- Scope: INSTANCE
- Default value: 60
- Range: [10, 2147483647]
- Unit: Seconds
- This variable is used to set the threshold value that determines whether to print expensive query logs. The difference between expensive query logs and slow query logs is:
 - Slow logs are printed after the statement is executed.
 - Expensive query logs print the statements that are being executed, with execution time exceeding the threshold value, and their related information.

11.10.1.107 tidb_force_priority

- Scope: INSTANCE
- Default value: NO_PRIORITY
- This variable is used to change the default priority for statements executed on a TiDB server. A use case is to ensure that a particular user that is performing OLAP queries receives lower priority than users performing OLTP queries.
- You can set the value of this variable to NO_PRIORITY, LOW_PRIORITY, DELAYED or HIGH_PRIORITY.

11.10.1.108 tidb_gc_concurrency New in v5.0

- Scope: GLOBAL
- Default value: -1
- Range: [1, 256]
- Unit: Threads
- Specifies the number of threads in the [Resolve Locks](#) step of GC. A value of -1 means that TiDB will automatically decide the number of garbage collection threads to use.

11.10.1.109 tidb_gc_enable New in v5.0

- Scope: GLOBAL
- Default value: ON
- Enables garbage collection for TiKV. Disabling garbage collection will reduce system performance, as old versions of rows will no longer be purged.

11.10.1.110 `tidb_gc_life_time` New in v5.0

- Scope: GLOBAL
- Default value: 10m0s
- Range: [10m0s, 8760h0m0s]
- The time limit during which data is retained for each GC, in the format of Go Duration. When a GC happens, the current time minus this value is the safe point.

Note:

- In scenarios of frequent updates, a large value (days or even months) for `tidb_gc_life_time` may cause potential issues, such as:
 - Larger storage use
 - A large amount of history data may affect performance to a certain degree, especially for range queries such as `select count(*) from t`
- If there is any transaction that has been running longer than `tidb_gc_life_time`, during GC, the data since `start_ts` is retained for this transaction to continue execution. For example, if `tidb_gc_life_time` is configured to 10 minutes, among all transactions being executed, the transaction that starts earliest has been running for 15 minutes, GC will retain data of the recent 15 minutes.

11.10.1.111 `tidb_gc_run_interval` New in v5.0

- Scope: GLOBAL
- Default value: 10m0s
- Range: [10m0s, 8760h0m0s]
- Specifies the GC interval, in the format of Go Duration, for example, "1h30m", and "15m"

11.10.1.112 `tidb_gc_scan_lock_mode` New in v5.0

Warning:

Currently, Green GC is an experimental feature. It is not recommended that you use it in production environments.

- Scope: GLOBAL
- Default value: LEGACY
- Possible values: PHYSICAL, LEGACY
 - LEGACY: Uses the old way of scanning, that is, disable Green GC.
 - PHYSICAL: Uses the physical scanning method, that is, enable Green GC.
- This variable specifies the way of scanning locks in the Resolve Locks step of GC. When the variable value is set to LEGACY, TiDB scans locks by Regions. When the value PHYSICAL is used, it enables each TiKV node to bypass the Raft layer and directly scan data, which can effectively mitigate the impact of GC wakening up all Regions when the **Hibernate Region** feature is enabled, thus improving the execution speed in the Resolve Locks step.

11.10.1.113 tidb_general_log

- Scope: INSTANCE
- Default value: OFF
- This variable is used to set whether to record all SQL statements in the **log**. This feature is disabled by default. If maintenance personnel needs to trace all SQL statements when locating issues, they can enable this feature.
- To see all records of this feature in the log, query the "GENERAL_LOG" string. The following information is recorded:
 - **conn**: The ID of the current session.
 - **user**: The current session user.
 - **schemaVersion**: The current schema version.
 - **txnStartTS**: The timestamp at which the current transaction starts.
 - **forUpdateTS**: In the pessimistic transactional mode, **forUpdateTS** is the current timestamp of the SQL statement. When a write conflict occurs in the pessimistic transaction, TiDB retries the SQL statement currently being executed and updates this timestamp. You can configure the number of retries via **max-retry-count**. In the optimistic transactional model, **forUpdateTS** is equivalent to **txnStartTS**.
 - **isReadConsistency**: Indicates whether the current transactional isolation level is Read Committed (RC).
 - **current_db**: The name of the current database.
 - **txn_mode**: The transactional mode. Value options are **OPTIMISTIC** and **PESSIMISTIC**.
 - **sql**: The SQL statement corresponding to the current query.

11.10.1.114 tidb_hash_join_concurrency

Warning:

Since v5.0, this variable is deprecated. Instead, use `tidb_executor_concurrency` for setting.

- Scope: SESSION | GLOBAL
- Default value: -1
- Range: [1, 256]
- Unit: Threads
- This variable is used to set the concurrency of the `hash join` algorithm.
- A value of -1 means that the value of `tidb_executor_concurrency` will be used instead.

11.10.1.115 `tidb_hashagg_final_concurrency`

Warning:

Since v5.0, this variable is deprecated. Instead, use `tidb_executor_concurrency` for setting.

- Scope: SESSION | GLOBAL
- Default value: -1
- Range: [1, 256]
- Unit: Threads
- This variable is used to set the concurrency of executing the concurrent hash → aggregation algorithm in the `final` phase.
- When the parameter of the aggregate function is not distinct, HashAgg is run concurrently and respectively in two phases - the `partial` phase and the `final` phase.
- A value of -1 means that the value of `tidb_executor_concurrency` will be used instead.

11.10.1.116 `tidb_hashagg_partial_concurrency`

Warning:

Since v5.0, this variable is deprecated. Instead, use `tidb_executor_concurrency` for setting.

- Scope: SESSION | GLOBAL
- Default value: -1
- Range: [1, 256]
- Unit: Threads
- This variable is used to set the concurrency of executing the concurrent hash ↵ aggregation algorithm in the partial phase.
- When the parameter of the aggregate function is not distinct, HashAgg is run concurrently and respectively in two phases - the partial phase and the final phase.
- A value of -1 means that the value of `tidb_executor_concurrency` will be used instead.

11.10.1.117 `tidb_index_join_batch_size`

- Scope: SESSION | GLOBAL
- Default value: 25000
- Range: [1, 2147483647]
- Unit: Rows
- This variable is used to set the batch size of the index lookup join operation.
- Use a bigger value in OLAP scenarios, and a smaller value in OLTP scenarios.

11.10.1.118 `tidb_index_lookup_concurrency`

Warning:

Since v5.0, this variable is deprecated. Instead, use `tidb_executor_concurrency` for setting.

- Scope: SESSION | GLOBAL
- Default value: -1
- Range: [1, 256]
- Unit: Threads
- This variable is used to set the concurrency of the index lookup operation.
- Use a bigger value in OLAP scenarios, and a smaller value in OLTP scenarios.
- A value of -1 means that the value of `tidb_executor_concurrency` will be used instead.

11.10.1.119 `tidb_index_lookup_join_concurrency`

Warning:

Since v5.0, this variable is deprecated. Instead, use [`tidb_executor_concurrency`](#) for setting.

- Scope: SESSION | GLOBAL
- Default value: -1
- Range: [1, 256]
- Unit: Threads
- This variable is used to set the concurrency of the `index lookup join` algorithm.
- A value of -1 means that the value of `tidb_executor_concurrency` will be used instead.

11.10.1.120 `tidb_index_lookup_size`

- Scope: SESSION | GLOBAL
- Default value: 20000
- Range: [1, 2147483647]
- Unit: Rows
- This variable is used to set the batch size of the `index lookup` operation.
- Use a bigger value in OLAP scenarios, and a smaller value in OLTP scenarios.

11.10.1.121 `tidb_index_serial_scan_concurrency`

- Scope: SESSION | GLOBAL
- Default value: 1
- Range: [1, 256]
- Unit: Threads
- This variable is used to set the concurrency of the `serial scan` operation.
- Use a bigger value in OLAP scenarios, and a smaller value in OLTP scenarios.

11.10.1.122 `tidb_init_chunk_size`

- Scope: SESSION | GLOBAL
- Default value: 32
- Range: [1, 32]
- Unit: Rows
- This variable is used to set the number of rows for the initial chunk during the execution process.

11.10.1.123 `tidb_isolation_read_engines` New in v4.0

- Scope: SESSION
- Default value: `tikv,tiflash,tidb`
- This variable is used to set the storage engine list that TiDB can use when reading data.

11.10.1.124 `tidb_log_file_max_days` New in v5.3

- Scope: SESSION
- Default value: 0
- This variable is used to adjust the maximum days of logger on the current TiDB instance. Its value defaults to the value of the `max-days` configuration in the configuration file. Changing the variable value only affects the current TiDB instance. After TiDB is restarted, the variable value is reset and the configuration value is not affected.

11.10.1.125 `tidb_low_resolution_tso`

- Scope: SESSION
- Default value: OFF
- This variable is used to set whether to enable the low precision TSO feature. After this feature is enabled, new transactions use a timestamp updated every 2 seconds to read data.
- The main applicable scenario is to reduce the overhead of acquiring TSO for small read-only transactions when reading old data is acceptable.

11.10.1.126 `tidb_max_chunk_size`

- Scope: SESSION | GLOBAL
- Default value: 1024
- Range: [32, 2147483647]
- Unit: Rows
- This variable is used to set the maximum number of rows in a chunk during the execution process. Setting to too large of a value may cause cache locality issues.

11.10.1.127 `tidb_max_delta_schema_count` New in v2.1.18 and v3.0.5

- Scope: GLOBAL
- Default value: 1024
- Range: [100, 16384]
- This variable is used to set the maximum number of schema versions (the table IDs modified for corresponding versions) allowed to be cached. The value range is 100 ~ 16384.

11.10.1.128 `tidb_mem_quota_apply_cache` New in v5.0

- Scope: SESSION | GLOBAL
- Default value: 33554432 (32 MiB)
- Range: [0, 9223372036854775807]
- Unit: Bytes
- This variable is used to set the memory usage threshold of the local cache in the `Apply` operator.
- The local cache in the `Apply` operator is used to speed up the computation of the `Apply` operator. You can set the variable to 0 to disable the `Apply` cache feature.

11.10.1.129 `tidb_mem_quota_query`

- Scope: SESSION
- Default value: 1073741824 (1 GiB)
- Range: [-1, 9223372036854775807]
- Unit: Bytes
- This variable is used to set the threshold value of memory quota for a query.
- If the memory quota of a query during execution exceeds the threshold value, TiDB performs the operation designated by the `OOMAction` option in the configuration file. The initial value of this variable is configured by `mem-quota-query`.

11.10.1.130 `tidb_memory_usage_alarm_ratio`

- Scope: INSTANCE
- Default value: 0.8
- TiDB triggers an alarm when the percentage of the memory it takes exceeds a certain threshold. For the detailed usage description of this feature, see `memory-usage-alarm-ratio`.
- You can set the initial value of this variable by configuring `memory-usage-alarm-ratio`.

11.10.1.131 `tidb_metric_query_range_duration` New in v4.0

- Scope: SESSION
- Default value: 60
- Range: [10, 216000]
- Unit: Seconds
- This variable is used to set the range duration of the Prometheus statement generated when querying `METRICS_SCHEMA`.

11.10.1.132 tidb_metric_query_step New in v4.0

- Scope: SESSION
- Default value: 60
- Range: [10, 216000]
- Unit: Seconds
- This variable is used to set the step of the Prometheus statement generated when querying METRICS_SCHEMA.

11.10.1.133 tidb_multi_statement_mode New in v4.0.11

- Scope: SESSION | GLOBAL
- Default value: OFF
- Possible values: OFF, ON, WARN
- This variable controls whether to allow multiple queries to be executed in the same COM_QUERY call.
- To reduce the impact of SQL injection attacks, TiDB now prevents multiple queries from being executed in the same COM_QUERY call by default. This variable is intended to be used as part of an upgrade path from earlier versions of TiDB. The following behaviors apply:

Client setting	tidb_multi_statement_mode value	Multiple statements permitted?
Multiple Statements = ON	OFF	Yes
Multiple Statements = ON	ON	Yes
Multiple Statements = ON	WARN	Yes
Multiple Statements = OFF	OFF	No
Multiple Statements = OFF	ON	Yes
Multiple Statements = OFF	WARN	Yes (+warning returned)

Note:

Only the default value of OFF can be considered safe. Setting tidb_multi_statement_mode=ON might be required if your application was specifically designed for an earlier version of TiDB. If your application requires multiple statement support, it is recommended to use the setting provided by your client library instead of the tidb_multi_statement_mode option. For example:

- go-sql-driver (`multiStatements`)
- Connector/J (`allowMultiQueries`)
- PHP mysqli (`mysqli_multi_query`)

11.10.1.134 tidb_opt_agg_push_down

- Scope: SESSION
- Default value: OFF
- This variable is used to set whether the optimizer executes the optimization operation of pushing down the aggregate function to the position before Join, Projection, and UnionAll.
- When the aggregate operation is slow in query, you can set the variable value to ON.

11.10.1.135 tidb_opt_correlation_exp_factor

- Scope: SESSION | GLOBAL
- Default value: 1
- Range: [0, 2147483647]
- When the method that estimates the number of rows based on column order correlation is not available, the heuristic estimation method is used. This variable is used to control the behavior of the heuristic method.
 - When the value is 0, the heuristic method is not used.
 - When the value is greater than 0:
 - * A larger value indicates that an index scan will probably be used in the heuristic method.
 - * A smaller value indicates that a table scan will probably be used in the heuristic method.

11.10.1.136 tidb_opt_correlation_threshold

- Scope: SESSION | GLOBAL
- Default value: 0.9
- This variable is used to set the threshold value that determines whether to enable estimating the row count by using column order correlation. If the order correlation between the current column and the `handle` column exceeds the threshold value, this method is enabled.

11.10.1.137 tidb_opt_distinct_agg_push_down

- Scope: SESSION
- Default value: OFF
- This variable is used to set whether the optimizer executes the optimization operation of pushing down the aggregate function with `distinct` (such as `select count(distinct a) from t`) to Coprocessor.
- When the aggregate function with the `distinct` operation is slow in the query, you can set the variable value to 1.

In the following example, before `tidb_opt_distinct_agg_push_down` is enabled, TiDB needs to read all data from TiKV and execute `distinct` on the TiDB side. After `tidb_opt_distinct_agg_push_down` is enabled, `distinct a` is pushed down to Coprocessor, and a group by column `test.t.a` is added to `HashAgg_5`.

```
mysql> desc select count(distinct a) from test.t;
+----+-----+-----+-----+-----+
| id | estRows | task | access object | operator info |
+----+-----+-----+-----+-----+
| StreamAgg_6 | 1.00 | root | funcs:count( | 
|   <-- distinct test.t.a)->Column#4 | 
| -TableReader_10 | 10000.00 | root | data: | 
|   <-- TableFullScan_9 | 
|   -TableFullScan_9 | 10000.00 | cop[tikv] | table:t | keep order:false |
|   , stats:pseudo | 
+----+-----+-----+-----+-----+
3 rows in set (0.01 sec)

mysql> set session tidb_opt_distinct_agg_push_down = 1;
Query OK, 0 rows affected (0.00 sec)

mysql> desc select count(distinct a) from test.t;
+----+-----+-----+-----+-----+
| id | estRows | task | access object | operator info |
+----+-----+-----+-----+-----+
| HashAgg_8 | 1.00 | root | funcs:count( | 
|   <-- distinct test.t.a)->Column#3 | 
| -TableReader_9 | 1.00 | root | data:HashAgg_5 | 
|   <-- HashAgg_5 | 
|   -TableFullScan_7 | 10000.00 | cop[tikv] | group by:test.t.a, | 
|   , stats:pseudo | 
+----+-----+-----+-----+-----+
```

```
+--+
→ -----+-----+-----+-----+
→
4 rows in set (0.00 sec)
```

11.10.1.138 tidb_opt_enable_correlation_adjustment

- Scope: SESSION | GLOBAL
- Default value: ON
- This variable is used to control whether the optimizer estimates the number of rows based on column order correlation

11.10.1.139 tidb_opt_insubq_to_join_and_agg

- Scope: SESSION | GLOBAL
- Default value: ON
- This variable is used to set whether to enable the optimization rule that converts a subquery to join and aggregation.
- For example, after you enable this optimization rule, the subquery is converted as follows:

```
select * from t where t.a in (select aa from t1);
```

The subquery is converted to join as follows:

```
select t.* from t, (select aa from t1 group by aa) tmp_t where t.a =
→ tmp_t.aa;
```

If `t1` is limited to be unique and not null in the `aa` column. You can use the following statement, without aggregation.

```
select t.* from t, t1 where t.a=t1.aa;
```

11.10.1.140 tidb_opt_limit_push_down_threshold

- Scope: SESSION | GLOBAL
- Default value: 100
- Range: [0, 2147483647]
- This variable is used to set the threshold that determines whether to push the Limit or TopN operator down to TiKV.
- If the value of the Limit or TopN operator is smaller than or equal to this threshold, these operators are forcibly pushed down to TiKV. This variable resolves the issue that the Limit or TopN operator cannot be pushed down to TiKV partly due to wrong estimation.

11.10.1.141 tidb_opt_prefer_range_scan New in v5.0

- Scope: SESSION | GLOBAL
- Default value: OFF
- After you set the value of this variable to ON, the optimizer always prefers range scans over full table scans.
- In the following example, before you enable `tidb_opt_prefer_range_scan`, the TiDB optimizer performs a full table scan. After you enable `tidb_opt_prefer_range_scan`, the optimizer selects an index range scan.

```
explain select * from t where age=5;
+---+
| id           | estRows | task      | access object | operator info
|             |          |          |              |
+---+
| TableReader_7 | 1048576.00 | root     |               | data:
|   ↳ Selection_6 |
| -Selection_6  | 1048576.00 | cop[tikv] |               | eq(test.t.age,
|   ↳ 5) |
|   -TableFullScan_5 | 1048576.00 | cop[tikv] | table:t | keep order:
|     ↳ false |
+---+
|             |          |          |              |
|             |          |          |              |
3 rows in set (0.00 sec)

set session tidb_opt_prefer_range_scan = 1;

explain select * from t where age=5;
+---+
| id           | estRows | task      | access object
|             |          |          | operator info
|             |          |          |
+---+
| IndexLookUp_7 | 1048576.00 | root     |               |
|   ↳           |          |          |               |
| -IndexRangeScan_5(Build) | 1048576.00 | cop[tikv] | table:t, index:
|   ↳ idx_age(age) | range:[5,5], keep order:false |
+---+
```

```
| -TableRowIDScan_6(Probe) | 1048576.00 | cop[tikv] | table:t
  ↵                      | keep order:false      |
+--+
  ↵ -----+-----+-----+
  ↵
3 rows in set (0.00 sec)
```

11.10.1.142 tidb_opt_write_row_id

- Scope: SESSION
- Default value: OFF
- This variable is used to control whether to allow INSERT, REPLACE, and UPDATE statements to operate on the `_tidb_rowid` column. This variable can be used only when you import data using TiDB tools.

11.10.1.143 tidb_partition_prune_mode New in v5.1

Warning:

Currently, the dynamic pruning mode for partitioned tables is an experimental feature. It is not recommended that you use it in production environments.

- Scope: SESSION | GLOBAL
- Default value: static
- Specifies whether to enable dynamic mode for partitioned tables. For details about the dynamic pruning mode, see [Dynamic Pruning Mode for Partitioned Tables](#).

11.10.1.144 tidb_pprof_sql_cpu New in v4.0

- Scope: INSTANCE
- Default value: 0
- Range: [0, 1]
- This variable is used to control whether to mark the corresponding SQL statement in the profile output to identify and troubleshoot performance issues.

11.10.1.145 tidb_projectionConcurrency

Warning:

Since v5.0, this variable is deprecated. Instead, use `tidb_executor_concurrency` for setting.

- Scope: SESSION | GLOBAL
- Default value: -1
- Range: [-1, 256]
- Unit: Threads
- This variable is used to set the concurrency of the Projection operator.
- A value of -1 means that the value of `tidb_executor_concurrency` will be used instead.

11.10.1.146 tidb_query_log_max_len

- Scope: INSTANCE
- Default value: 4096 (4 KiB)
- Range: [-1, 9223372036854775807]
- Unit: Bytes
- The maximum length of the SQL statement output. When the output length of a statement is larger than the `tidb_query-log-max-len` value, the statement is truncated to output.

Usage example:

```
SET tidb_query_log_max_len = 20
```

11.10.1.147 tidb_record_plan_in_slow_log

- Scope: INSTANCE
- Default value: ON
- This variable is used to control whether to include the execution plan of slow queries in the slow log.

11.10.1.148 tidb_redact_log

- Scope: SESSION | GLOBAL
- Default value: OFF

- This variable controls whether to hide user information in the SQL statement being recorded into the TiDB log and slow log.
- When you set the variable to 1, user information is hidden. For example, if the executed SQL statement is `insert into t values (1,2)`, the statement is recorded as `insert ↪ into t values (?,?)` in the log.

11.10.1.149 `tidb_replica_read` New in v4.0

- Scope: SESSION | GLOBAL
- Default value: `leader`
- Possible values: `leader`, `follower`, `leader-and-follower`, `closest-relicas`
- This variable is used to control where TiDB reads data. Here are three options:
 - `leader`: Read only from leader node
 - `follower`: Read only from follower node
 - `leader-and-follower`: Read from leader or follower node
- See [follower reads](#) for additional details.

11.10.1.150 `tidb_retry_limit`

- Scope: SESSION | GLOBAL
- Default value: 10
- Range: `[-1, 9223372036854775807]`
- This variable is used to set the maximum number of the retries for optimistic transactions. When a transaction encounters retryable errors (such as transaction conflicts, very slow transaction commit, or table schema changes), this transaction is re-executed according to this variable. Note that setting `tidb_retry_limit` to 0 disables the automatic retry. This variable only applies to optimistic transactions, not to pessimistic transactions.

11.10.1.151 `tidb_row_format_version`

- Scope: GLOBAL
- Default value: 2
- Range: `[1, 2]`
- Controls the format version of the newly saved data in the table. In TiDB v4.0, the [new storage row format](#) version 2 is used by default to save new data.
- If you upgrade from a TiDB version earlier than 4.0.0 to 4.0.0, the format version is not changed, and TiDB continues to use the old format of version 1 to write data to the table, which means that **only newly created clusters use the new data format by default**.
- Note that modifying this variable does not affect the old data that has been saved, but applies the corresponding version format only to the newly written data after modifying this variable.

11.10.1.152 tidb_scatter_region

- Scope: GLOBAL
- Default value: OFF
- By default, Regions are split for a new table when it is being created in TiDB. After this variable is enabled, the newly split Regions are scattered immediately during the execution of the CREATE TABLE statement. This applies to the scenario where data need to be written in batches right after the tables are created in batches, because the newly split Regions can be scattered in TiKV beforehand and do not have to wait to be scheduled by PD. To ensure the continuous stability of writing data in batches, the CREATE TABLE statement returns success only after the Regions are successfully scattered. This makes the statement's execution time multiple times longer than that when you disable this variable.

11.10.1.153 tidb_skip_ascii_check New in v5.0

- Scope: SESSION | GLOBAL
- Default value: OFF
- This variable is used to set whether to skip ASCII validation.
- Validating ASCII characters affects the performance. When you are sure that the input characters are valid ASCII characters, you can set the variable value to ON.

11.10.1.154 tidb_skip_isolation_level_check

- Scope: SESSION | GLOBAL
- Default value: OFF
- After this switch is enabled, if an isolation level unsupported by TiDB is assigned to tx_isolation, no error is reported. This helps improve compatibility with applications that set (but do not depend on) a different isolation level.

```

tidb> set tx_isolation='Serializable';
ERROR 8048 (HY000): The isolation level 'Serializable' is not supported.
    ↪ Set tidb_skip_isolation_level_check=1 to skip this error
tidb> set tidb_skip_isolation_level_check=1;
Query OK, 0 rows affected (0.00 sec)

tidb> set tx_isolation='Serializable';
Query OK, 0 rows affected, 1 warning (0.00 sec)

```

11.10.1.155 tidb_skip_utf8_check

- Scope: SESSION | GLOBAL
- Default value: OFF
- This variable is used to set whether to skip UTF-8 validation.
- Validating UTF-8 characters affects the performance. When you are sure that the input characters are valid UTF-8 characters, you can set the variable value to ON.

11.10.1.156 tidb_slow_log_threshold

- Scope: INSTANCE
- Default value: 300
- Range: [-1, 9223372036854775807]
- Unit: Milliseconds
- This variable is used to output the threshold value of the time consumed by the slow log. When the time consumed by a query is larger than this value, this query is considered as a slow log and its log is output to the slow query log.

Usage example:

```
SET tidb_slow_log_threshold = 200;
```

11.10.1.157 tidb_slow_query_file

- Scope: SESSION
- Default value: “”
- When INFORMATION_SCHEMA.SLOW_QUERY is queried, only the slow query log name set by slow-query-file in the configuration file is parsed. The default slow query log name is “tidb-slow.log”. To parse other logs, set the tidb_slow_query_file session variable to a specific file path, and then query INFORMATION_SCHEMA.SLOW_QUERY to parse the slow query log based on the set file path. For details, see [Identify Slow Queries](#).

11.10.1.158 tidb_snapshot

- Scope: SESSION
- Default value: “”
- This variable is used to set the time point at which the data is read by the session. For example, when you set the variable to “2017-11-11 20:20:20” or a TSO number like “400036290571534337”, the current session reads the data of this moment.

11.10.1.159 tidb_stmt_summary_history_size New in v4.0

- Scope: SESSION | GLOBAL
- Default value: 24
- Range: [0, 255]
- This variable is used to set the history capacity of [statement summary tables](#).

11.10.1.160 tidb_stmt_summary_internal_query New in v4.0

- Scope: SESSION | GLOBAL
- Default value: OFF
- This variable is used to control whether to include the SQL information of TiDB in [statement summary tables](#).

11.10.1.161 tidb_stmt_summary_max_sql_length New in v4.0

- Scope: SESSION | GLOBAL
- Default value: 4096
- Range: [0, 2147483647]
- This variable is used to control the length of the SQL string in [statement summary tables](#).

11.10.1.162 tidb_stmt_summary_max_stmt_count New in v4.0

- Scope: SESSION | GLOBAL
- Default value: 3000
- Range: [1, 32767]
- This variable is used to set the maximum number of statements that [statement summary tables](#) store in memory.

11.10.1.163 tidb_stmt_summary_refresh_interval New in v4.0

- Scope: SESSION | GLOBAL
- Default value: 1800
- Range: [1, 2147483647]
- Unit: Seconds
- This variable is used to set the refresh time of [statement summary tables](#).

11.10.1.164 tidb_store_limit New in v3.0.4 and v4.0

- Scope: INSTANCE | GLOBAL
- Default value: 0
- Range: [0, 9223372036854775807]
- This variable is used to limit the maximum number of requests TiDB can send to TiKV at the same time. 0 means no limit.

11.10.1.165 tidb_tmp_table_max_size New in v5.3.0

- Scope: SESSION | GLOBAL
- Default value: 67108864
- Range: [1048576, 137438953472]
- Unit: Bytes
- This variable is used to set the maximum size of a single temporary table. Any temporary table with a size larger than this variable value causes error.

11.10.1.166 tidb_tso_client_batch_max_wait_time New in v5.3

- Scope: GLOBAL
- Default value: 0
- Range: [0, 10]
- Unit: Milliseconds
- This variable is used to set the maximum waiting time for a batch operation when TiDB requests TSO from PD. The default value is 0, which means no extra waiting time.
- When obtaining TSO requests from PD each time, PD Client, used by TiDB, collects as many TSO requests received at the same time as possible. Then, PD Client merges the collected requests in batch into one RPC request and sends the request to PD. This helps reduce the pressure on PD.
- After setting this variable to a value greater than 0, TiDB waits for the maximum duration of this value before the end of each batch merge. This is to collect more TSO requests and improve the effect of batch operations.
- Scenarios for increasing the value of this variable:
 - Due to the high pressure of TSO requests, the CPU of the PD leader reaches a bottleneck, which causes high latency of TSO RPC requests.
 - There are not many TiDB instances in the cluster, but every TiDB instance is in high concurrency.
- It is recommended to set this variable to a value as small as possible.

Notes:

Suppose that the TSO RPC latency increases for reasons other than a CPU usage bottleneck of the PD leader (such as network issues). In this case, increasing the value of `tidb_tso_client_batch_max_wait_time` might increase the execution latency in TiDB and affect the QPS performance of the cluster.

11.10.1.167 tidb_txn_mode

- Scope: SESSION | GLOBAL
- Default value: pessimistic
- Possible values: `pessimistic`, `optimistic`
- This variable is used to set the transaction mode. TiDB 3.0 supports the pessimistic transactions. Since TiDB 3.0.8, the `pessimistic transaction mode` is enabled by default.
- If you upgrade TiDB from v3.0.7 or earlier versions to v3.0.8 or later versions, the default transaction mode does not change. **Only the newly created clusters use the pessimistic transaction mode by default.**
- If this variable is set to “optimistic” or “”, TiDB uses the `optimistic transaction mode`.

11.10.1.168 tidb_use_plan_baselines New in v4.0

- Scope: SESSION | GLOBAL
- Default value: ON
- This variable is used to control whether to enable the execution plan binding feature. It is enabled by default, and can be disabled by assigning the OFF value. For the use of the execution plan binding, see [Execution Plan Binding](#).

11.10.1.169 tidb_wait_split_region_finish

- Scope: SESSION
- Default value: ON
- It usually takes a long time to scatter Regions, which is determined by PD scheduling and TiKV loads. This variable is used to set whether to return the result to the client after all Regions are scattered completely when the `SPLIT REGION` statement is being executed:
 - ON requires that the `SPLIT REGIONS` statement waits until all Regions are scattered.
 - OFF permits the `SPLIT REGIONS` statement to return before finishing scattering all Regions.
- Note that when scattering Regions, the write and read performances for the Region that is being scattered might be affected. In batch-write or data importing scenarios, it is recommended to import data after Regions scattering is finished.

11.10.1.170 tidb_wait_split_region_timeout

- Scope: SESSION
- Default value: 300
- Range: [1, 2147483647]
- Unit: Seconds

- This variable is used to set the timeout for executing the `SPLIT REGION` statement. If a statement is not executed completely within the specified time value, a timeout error is returned.

11.10.1.171 `tidb_window_concurrency` New in v4.0

Warning:

Since v5.0, this variable is deprecated. Instead, use `tidb_executor_concurrency` for setting.

- Scope: SESSION | GLOBAL
- Default value: -1
- Range: [1, 256]
- Unit: Threads
- This variable is used to set the concurrency degree of the window operator.
- A value of -1 means that the value of `tidb_executor_concurrency` will be used instead.

11.10.1.172 `time_zone`

- Scope: SESSION | GLOBAL
- Default value: SYSTEM
- This variable returns the current time zone. Values can be specified as either an offset such as '-8:00' or a named zone 'America/Los_Angeles'.
- The value SYSTEM means that the time zone should be the same as the system host, which is available via the `system_time_zone` variable.

11.10.1.173 `timestamp`

- Scope: SESSION
- Default value: “”
- A non-empty value of this variable indicates the UNIX epoch that is used as the timestamp for `CURRENT_TIMESTAMP()`, `NOW()`, and other functions. This variable might be used in data restore or replication.

11.10.1.174 `transaction_isolation`

- Scope: SESSION | GLOBAL
- Default value: REPEATABLE-READ

- Possible values: READ-UNCOMMITTED, READ-COMMITTED, REPEATABLE-READ, SERIALIZABLE
↪
- This variable sets the transaction isolation. TiDB advertises REPEATABLE-READ for compatibility with MySQL, but the actual isolation level is Snapshot Isolation. See [transaction isolation levels](#) for further details.

11.10.1.175 tx_isolation

This variable is an alias for `transaction_isolation`.

11.10.1.176 version

- Scope: NONE
- Default value: 5.7.25-TiDB-(tidb version)
- This variable returns the MySQL version, followed by the TiDB version. For example ‘5.7.25-TiDB-v4.0.0-beta.2-716-g25e003253’.

11.10.1.177 version_comment

- Scope: NONE
- Default value: (string)
- This variable returns additional details about the TiDB version. For example, ‘TiDB Server (Apache License 2.0) Community Edition, MySQL 5.7 compatible’.

11.10.1.178 version_compile_os

- Scope: NONE
- Default value: (string)
- This variable returns the name of the OS on which TiDB is running.

11.10.1.179 version_compile_machine

- Scope: NONE
- Default value: (string)
- This variable returns the name of the CPU architecture on which TiDB is running.

11.10.1.180 wait_timeout

- Scope: SESSION | GLOBAL
- Default value: 0
- Range: [0, 31536000]
- Unit: Seconds
- This variable controls the idle timeout of user sessions. A zero-value means unlimited.

11.10.1.181 warning_count

- Scope: SESSION
- Default value: 0
- This read-only variable indicates the number of warnings that occurred in the statement that was previously executed.

11.10.1.182 windowing_use_high_precision

- Scope: SESSION | GLOBAL
- Default value: ON
- This variable controls whether to use the high precision mode when computing the window functions.

11.11 Storage Engines

11.11.1 TiKV

11.11.1.1 TiKV Overview

TiKV is a distributed and transactional key-value database, which provides transactional APIs with ACID compliance. With the implementation of the [Raft consensus algorithm](#) and consensus state stored in RocksDB, TiKV guarantees data consistency between multiple replicas and high availability. As the storage layer of the TiDB distributed database, TiKV provides the read and write service, and persist the written data from applications. It also stores the statistics data of the TiDB cluster.

11.11.1.1.1 Architecture Overview

TiKV implements the multi-raft-group replica mechanism based on the design of Google Spanner. A Region is a basic unit of the key-value data movement and refers to a data range in a Store. Each Region is replicated to multiple nodes. These multiple replicas form a Raft group. A replica of a Region is called a Peer. Typically there are 3 peers in a Region. One of them is the leader, which provides the read and write services. The PD component balances all the Regions automatically to guarantee that the read and write throughput is balanced among all the nodes in the TiKV cluster. With PD and carefully designed Raft groups, TiKV excels in horizontal scalability and can easily scale to store more than 100 TBs of data.

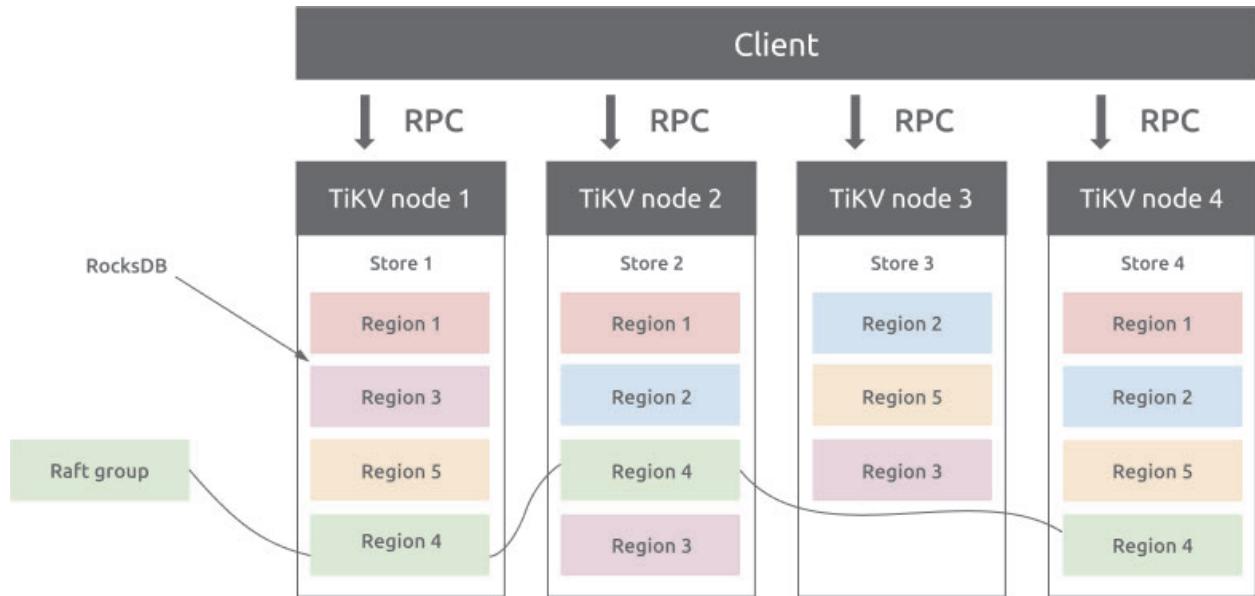


Figure 464: TiKV Architecture

Region and RocksDB

There is a RocksDB database within each Store and it stores data into the local disk. All the Region data are stored in the same RocksDB instance in each Store. All the logs used for the Raft consensus algorithm is stored in another RocksDB instance in each Store. This is because the performance of sequential I/O is better than random I/O. With different RocksDB instances storing raft logs and Region data, TiKV combines all the data write operations of raft logs and TiKV Regions into one I/O operation to improve the performance.

Region and Raft Consensus Algorithm

Data consistency between replicas of a Region is guaranteed by the Raft Consensus Algorithm. Only the leader of the Region can provide the writing service, and only when the data is written to the majority of replicas of a Region, the write operation succeeds.

When the size of a Region exceeds a threshold, which is 144 MB by default, TiKV splits it to two or more Regions. This operation guarantees the size of all the Regions in the cluster is nearly the same, which helps the PD component to balance Regions among nodes in a TiKV cluster. When the size of a Region is smaller than the threshold, TiKV merges the two smaller adjacent Regions into one Region.

When PD moves a replica from one TiKV node to another, it firstly adds a Learner replica on the target node, after the data in the Learner replica is nearly the same as that in the Leader replica, PD changes it to a Follower replica and removes the Follower replica on the source node.

Moving Leader replica from one node to another has a similar mechanism. The difference is that after the Learner replica becomes the Follower replica, there is a “Leader Transfer”

operation in which the Follower replica actively proposes an election to elect itself as the Leader. Finally, the new Leader removes the old Leader replica in the source node.

11.11.1.1.2 Distributed Transaction

TiKV supports distributed transactions. Users (or TiDB) can write multiple key-value pairs without worrying about whether they belong to the same Region. TiKV uses two-phase commit to achieve ACID constraints. See [TiDB Optimistic Transaction Model](#) for details.

11.11.1.1.3 TiKV Coprocessor

TiDB pushes some data computation logic to TiKV Coprocessor. TiKV Coprocessor processes the computation for each Region. Each request sent to TiKV Coprocessor only involves the data of one Region.

11.11.1.2 RocksDB Overview

[RocksDB](#) is an LSM-tree storage engine that provides key-value store and read-write functions. It is developed by Facebook and based on LevelDB. Key-value pairs written by the user are firstly inserted into Write Ahead Log (WAL) and then written to the SkipList in memory (a data structure called MemTable). LSM-tree engines convert the random modification (insertion) to sequential writes to the WAL file, so they provide better write throughput than B-tree engines.

Once the data in memory reaches a certain size, RocksDB flushes the content into a Sorted String Table (SST) file in the disk. SST files are organized in multiple levels (the default is up to 6 levels). When the total size of a level reaches the threshold, RocksDB chooses part of the SST files and merges them into the next level. Each subsequent level is 10 times larger than the previous one, so 90% of the data is stored in the last layer.

RocksDB allows users to create multiple Column Families (CFs). CFs have their own SkipList and SST files, and they share the same WAL file. In this way, different CFs can have different settings according to the application characteristics. It does not increase the number of writes to WAL at the same time.

11.11.1.2.1 TiKV architecture

The architecture of TiKV is illustrated as follows:

TiKV Architecture

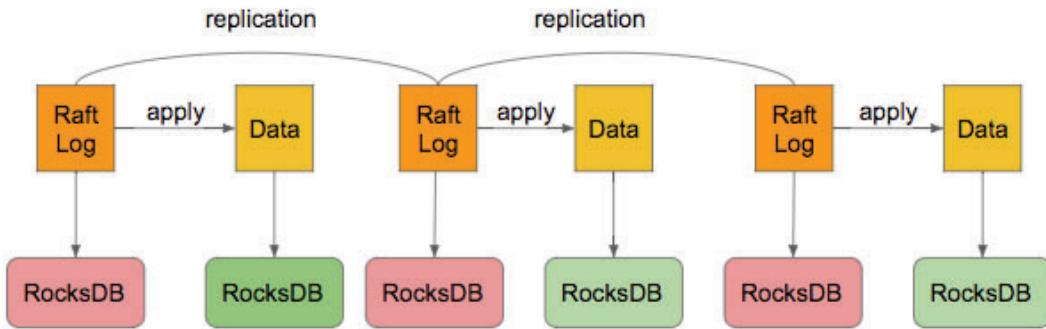


Figure 465: TiKV RocksDB

As the storage engine of TiKV, RocksDB is used to store Raft logs and user data. All data in a TiKV node shares two RocksDB instances. One is for Raft log (often called raftdb), and the other is for user data and MVCC metadata (often called kvdb). There are four CFs in kvdb: raft, lock, default, and write:

- raft CF: Store metadata of each Region. It occupies only a very small amount of space, and users do not need to care.
- lock CF: Store the pessimistic lock of pessimistic transactions and the Prewrite lock for distributed transactions. After the transaction is committed, the corresponding data in lock CF is deleted quickly. Therefore, the size of data in lock CF is usually very small (less than 1 GB). If the data in lock CF increases a lot, it means that a large number of transactions are waiting to be committed, and that the system meets a bug or failure.
- write CF: Store the user's real written data and MVCC metadata (the start timestamp and commit timestamp of the transaction to which the data belongs). When the user writes a row of data, it is stored in the write CF if the data length is less than 255 bytes. Otherwise, it is stored in the default CF. In TiDB, the secondary index only occupies the space of write CF, since the value stored in the non-unique index is empty and the value stored in the unique index is the primary key index.
- default CF: Store data longer than 255 bytes.

11.11.1.2.2 RocksDB memory usage

To improve the reading performance and reduce the reading operations to the disk, RocksDB divides the files stored on the disk into blocks based on a certain size (the default

is 64 KB). When reading a block, it first checks if the data already exists in BlockCache in memory. If true, it can read the data directly from memory without accessing the disk.

BlockCache discards the least recently used data according to the LRU algorithm. By default, TiKV devotes 45% of the system memory to BlockCache. Users can also modify the `storage.block-cache.capacity` configuration to an appropriate value by themselves. However, it is not recommended to exceed 60% of the total system memory.

The data written to RocksDB is written to MemTable firstly. When the size of a MemTable exceeds 128 MB, it switches to a new MemTable. There are 2 RocksDB instances in TiKV, a total of 4 CFs. The size limit of a single MemTable for each CF is 128 MB. A maximum of 5 MemTables can exist at the same time; otherwise, the foreground writes is blocked. The memory occupied by this part is at most 2.5 GB ($4 \times 5 \times 128$ MB). It is not recommended to change this limit since it costs less memory.

11.11.1.2.3 RocksDB space usage

- Multi-version: As RocksDB is a key-value storage engine with LSM-tree structure, the data in MemTable is flushed to L0 first. Because the file is arranged in the order of which they are generated, there might be overlap between the ranges of SSTs at the L0. As a result, the same key might have multiple versions in L0. When a file is merged from L0 to L1, it is cut into multiple files in a certain size (the default is 8 MB). The key range of each file on the same level does not overlap with each other, so there is only one version for each key on L1 and subsequent levels.
- Space amplification: The total size of files on each level is x (the default is 10) times that of the previous level, so 90% of the data is stored in the last level. It also means that the space amplification of RocksDB does not exceed 1.11 (L0 has fewer data and can be ignored).
- Space amplification of TiKV: TiKV has its own MVCC strategy. When a user writes a key, the real data written to RocksDB is key + commit_ts, that is to say, the update and deletion also write a new key to RocksDB. TiKV deletes the old version of the data (through the Delete interface of RocksDB) at intervals, so it can be considered that the actual space of the data stored by the user on TiKV is enlarged to 1.11 plus the data written in the last 10 minutes (assuming that TiKV cleans up the old data promptly).

11.11.1.2.4 RocksDB background threads and compaction

In RocksDB, operations such as converting the MemTable into SST files and merging SST files at various levels are performed in the background thread pool. The default size of the background thread pool is 8. When the number of CPUs in the machine is less than or equal to 8, the default size of the background thread pool is the number of CPUs minus one.

Generally speaking, users do not need to change this configuration. If the user deploys multiple TiKV instances on a machine, or the machine has a relatively high read load and a low write load, you can adjust the `rocksdb/max-background-jobs` to 3 or 4 as appropriate.

11.11.1.2.5 WriteStall

The L0 of RocksDB is different from other levels. The SSTs of L0 are arranged in the order of generation. The key ranges between the SSTs can overlap. Therefore, each SST in L0 must be queried in turn when a query is performed. In order not to affect query performance, WriteStall is triggered to block writing when there are too many files in L0.

When encountering a sudden sharp increase in write delay, you can first check the **WriteStall Reason** metric on the Grafana RocksDB KV panel. If it is a WriteStall caused by too many L0 files, you can adjust the following configurations to 64.

```
rocksdb.defaultcf.level0-slowdown-writes-trigger
rocksdb.writecf.level0-slowdown-writes-trigger
rocksdb.lockcf.level0-slowdown-writes-trigger
rocksdb.defaultcf.level0-stop-writes-trigger
rocksdb.writecf.level0-stop-writes-trigger
rocksdb.lockcf.level0-stop-writes-trigger
```

11.11.1.3 Titan Overview

Titan is a high-performance [RocksDB](#) plugin for key-value separation. Titan can reduce write amplification in RocksDB when large values are used.

When the value size in Key-Value pairs is large, Titan performs better than RocksDB in write, update, and point read scenarios. However, Titan gets a higher write performance by sacrificing storage space and range query performance. As the price of SSDs continues to decrease, this trade-off will be more and more meaningful.

11.11.1.3.1 Key features

- Reduce write amplification by separating values from the log-structured merge-tree (LSM-tree) and storing them independently.
- Seamlessly upgrade RocksDB instances to Titan. The upgrade does not require human intervention and does not impact online services.
- Achieve 100% compatibility with all RocksDB features used by the current TiKV.

11.11.1.3.2 Usage scenarios

Titan is suitable for the scenarios where a huge volume of data is written to the TiKV foreground:

- RocksDB triggers a large amount of compactions, which consumes a lot of I/O bandwidth or CPU resources. This causes poor read and write performance of the foreground.
- The RocksDB compaction lags much behind (due to the I/O bandwidth limit or CPU bottleneck) and frequently causes write stalls.
- RocksDB triggers a large amount of compactions, which causes a lot of I/O writes and affects the life of the SSD disk.

11.11.1.3.3 Prerequisites

The prerequisites for enabling Titan are as follows:

- The average size of values is large, or the size of all large values accounts for much of the total value size. Currently, the size of a value greater than 1 KB is considered as a large value. In some situations, this number (1 KB) can be 512 B. Note that a single value written to TiKV cannot exceed 8 MB due to the limitation of the TiKV Raft layer. You can adjust the `raft-entry-max-size` configuration value to relax the limit.
- No range query will be performed or you do not need a high performance of range query. Because the data stored in Titan is not well-ordered, its performance of range query is poorer than that of RocksDB, especially for the query of a large range. According to PingCAP's internal test, Titan's range query performance is 40% to a few times lower than that of RocksDB.
- Sufficient disk space, because Titan reduces write amplification at the cost of disk space. In addition, Titan compresses values one by one, and its compression rate is lower than that of RocksDB. RocksDB compresses blocks one by one. Therefore, Titan consumes more storage space than RocksDB, which is expected and normal. In some situations, Titan's storage consumption can be twice that of RocksDB.

If you want to improve the performance of Titan, see the blog post [Titan: A RocksDB Plugin to Reduce Write Amplification](#).

11.11.1.3.4 Architecture and implementation

The following figure shows the architecture of Titan:

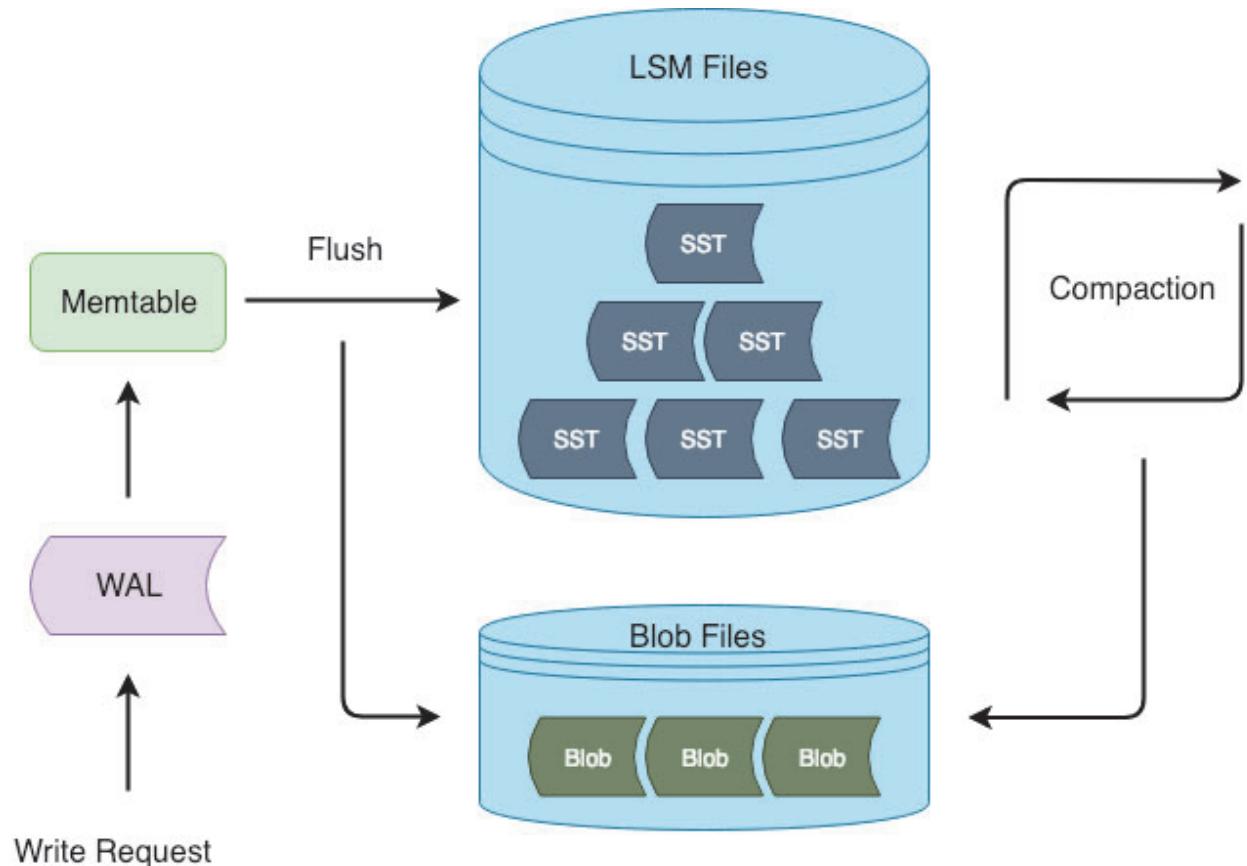


Figure 466: Titan Architecture

During flush and compaction operations, Titan separates values from the LSM-tree. The advantage of this approach is that the write process is consistent with RocksDB, which reduces the chance of invasive changes to RocksDB.

BlobFile

When Titan separates the value file from the LSM-tree, it stores the value file in the BlobFile. The following figure shows the BlobFile format:

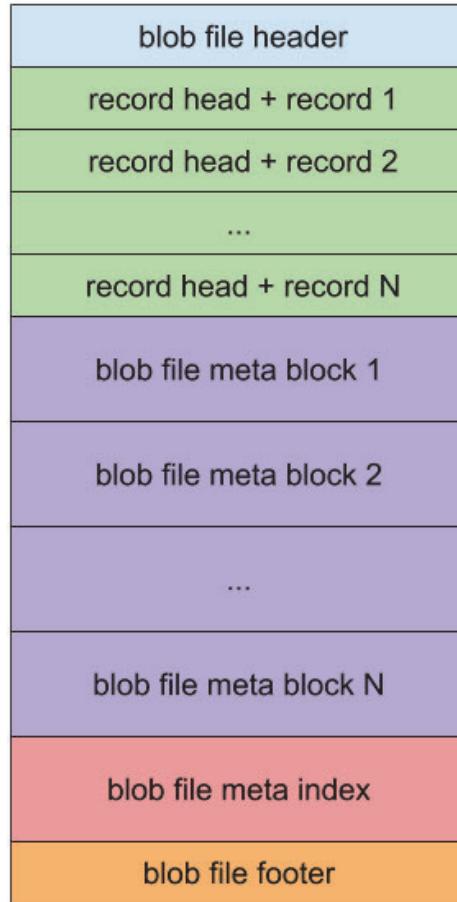


Figure 467: BlobFile Format

A blob file mainly consists of blob records, meta blocks, a meta index block, and a footer. Each block record stores a Key-Value pair. The meta blocks are used for scalability, and store properties related to the blob file. The meta index block is used for meta block searching.

Note:

- The Key-Value pairs in the blob file are stored in order, so that when the Iterator is implemented, the sequential reading performance can be improved via prefetching.
- Each blob record keeps a copy of the user key corresponding to the value. This way, when Titan performs Garbage Collection (GC), it can query the user key and identify whether the corresponding value is outdated. However, this process introduces some write amplification.

- BlobFile supports compression at the blob record level. Titan supports multiple compression algorithms, such as [Snappy](#), [LZ4](#), and [Zstd](#). Currently, the default compression algorithm Titan uses is LZ4.

TitanTableBuilder

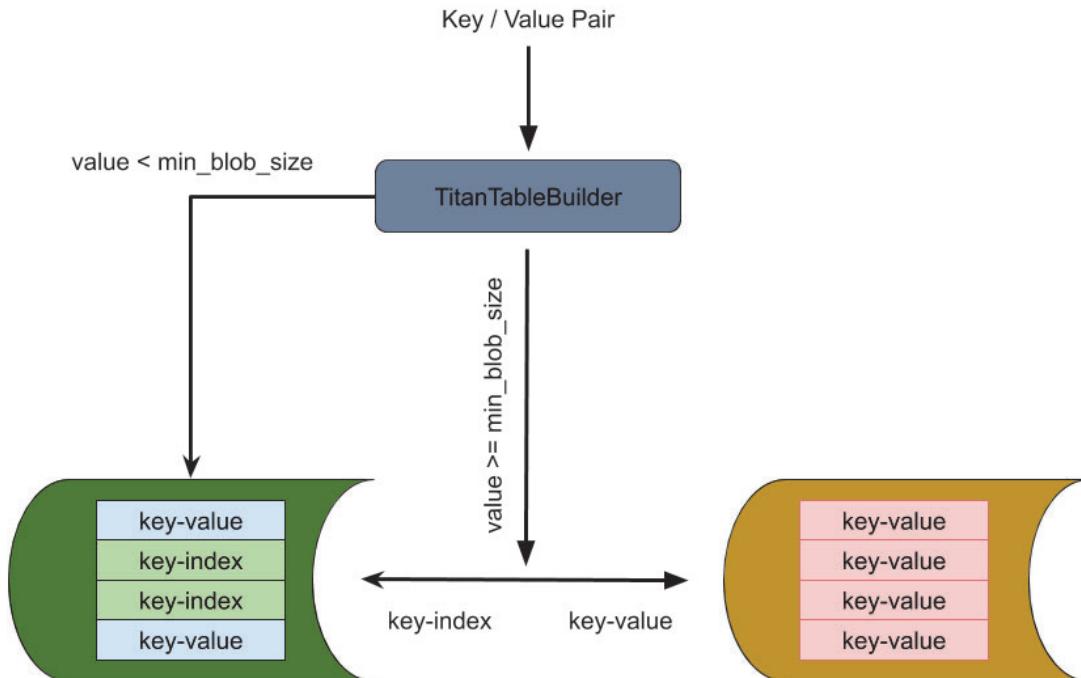


Figure 468: TitanTableBuilder

`TitanTableBuilder` is the key to achieving Key-Value separation. `TitanTableBuilder` determines the Key-Pair value size, and based on that, decides whether to separate the value from the Key-Value pair and store it in the blob file.

- If the value size is greater than or equal to `min_blob_size`, `TitanTableBuilder` separates the value and stores it in the blob file. `TitanTableBuilder` also generates an index and writes it into the SST.
- If the value size is smaller than `min_blob_size`, `TitanTableBuilder` writes the value directly into the SST.

Titan can also be downgraded to RocksDB in the process above. When RocksDB is performing compactions, the separated value can be written back to the newly generated SST files.

11.11.1.3.5 Garbage Collection

Titan uses Garbage Collection (GC) to reclaim space. As the keys are being reclaimed in the LSM-tree compaction, some values stored in blob files are not deleted at the same time. Therefore, Titan needs to perform GC periodically to delete outdated values. Titan provides the following two types of GC:

- Blob files are periodically integrated and rewritten to delete outdated values. This is the regular way of performing GC.
- Blob files are rewritten while the LSM-tree compaction is performed at the same time. This is the feature of Level Merge.

Regular GC

Titan uses the TablePropertiesCollector and EventListener components of RocksDB to collect the information for GC.

TablePropertiesCollector

RocksDB supports using BlobFileSizeCollector, a custom table property collector, to collect properties from the SST which are written into corresponding SST files. The collected properties are named BlobFileSizeProperties. The following figure shows the BlobFileSizeCollector workflow and data formats:

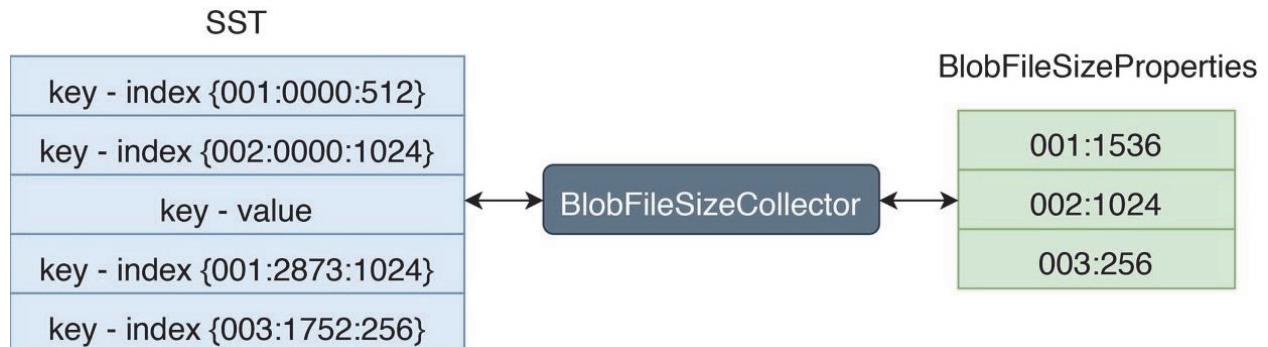


Figure 469: BlobFileSizeProperties

On the left is the SST index format. The first column is the blob file ID; the second column is the offset for the blob record in the blob file; the third column is the blob record size.

On the right is the BlobFileSizeProperties format. Each line represents a blob file and how much data is saved in this blob file. The first column is the blob file ID; the second column is the size of the data.

EventListener

RocksDB uses compaction to discard old data and reclaim space. After each compaction, some blob files in Titan might contain partly or entirely outdated data. Therefore, you can

trigger GC by listening to compaction events. During compaction, you can collect and compare the input/output blob file size properties of SST to determine which blob files require GC. The following figure shows the general process:

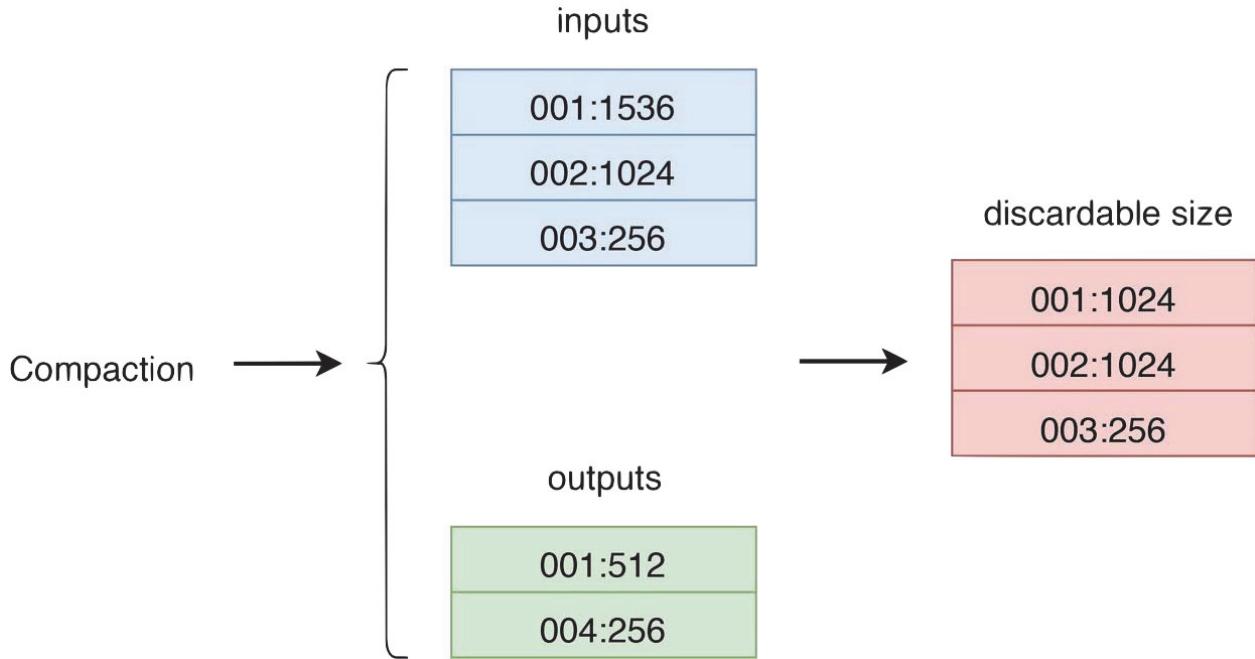


Figure 470: EventListener

- *inputs* stands for the blob file size properties for all SSTs that participate in the compaction.
- *outputs* stands for the blob file size properties for all SSTs generated in the compaction.
- *discardable size* is the size of the file to be discarded for each blob file, calculated based on inputs and outputs. The first column is the blob file ID. The second column is the size of the file to be discarded.

For each valid blob file, Titan maintains a discardable size variable in memory. After each compaction, this variable is accumulated for the corresponding blob file. Each time when GC starts, it picks the blob file with the greatest discardable size as the candidate file for GC. To reduce write amplification, a certain level of space amplification is allowed, which means GC can be started on a blob file only when the discardable file has reached a specific proportion in size.

For the selected blob file, Titan checks whether the blob index of the key corresponding to each value exists or has been updated to determine whether this value is outdated. If the value is not outdated, Titan merges and sorts the value into a new blob file, and writes the updated blob index into SST using WriteCallback or MergeOperator. Then, Titan records the latest sequence number of RocksDB and does not delete the old blob file until the sequence of the oldest snapshot exceeds the recorded sequence number. The reason is

that after the blob index is written back to SST, the old blob index is still accessible via the previous snapshot. Therefore, we need to ensure that no snapshot will access the old blob index before GC can safely delete the corresponding blob file.

Level Merge

Level Merge is a newly introduced algorithm in Titan. According to the implementation principle of Level Merge, Titan merges and rewrites blob file that corresponds to the SST file, and generates new blob file while compactions are performed in LSM-tree. The following figure shows the general process:

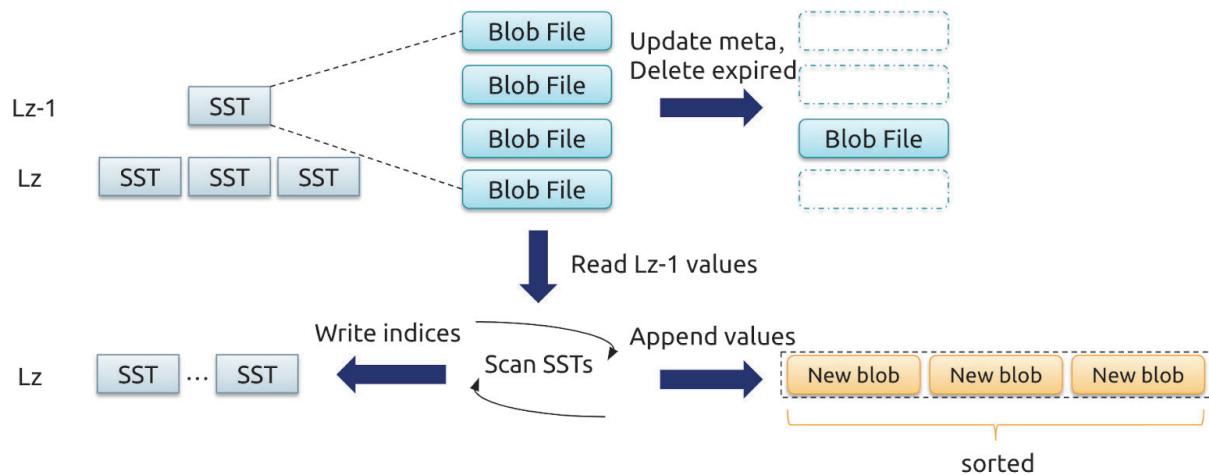


Figure 471: LevelMerge General Process

When compactions are performed on the SSTs of level $z-1$ and level z , Titan reads and writes Key-Value pairs in order. Then it writes the values of the selected blob files into new blob files in order, and updates the blob indexes of keys when new SSTs are generated. For the keys deleted in compactions, the corresponding values will not be written to the new blob file, which works similar to GC.

Compared with the regular way of GC, the Level Merge approach completes the blob GC while compactions are performed in LSM-tree. In this way, Titan no longer needs to check the status of blob index in LSM-tree or to write the new blob index into LSM-tree. This reduces the impact of GC on the foreground operations. As the blob file is repeatedly rewritten, fewer files overlap with each other, which makes the whole system in better order and improves the performance of scan.

However, layering blob files similar to tiering compaction brings write amplification. Because 99% of the data in LSM-tree is stored at the lowest two levels, Titan performs the Level Merge operation on the blob files corresponding to the data that is compacted only to the lowest two levels of LSM-tree.

Range Merge

Range Merge is an optimized approach of GC based on Level Merge. However, the bottom level of LSM-tree might be in poorer order in the following situations:

- When `level_compaction_dynamic_level_bytes` is enabled, data volume at each level of LSM-tree dynamically increases, and the sorted runs at the bottom level keep increasing.
- A specific range of data is frequently compacted, and this causes a lot of sorted runs in that range.

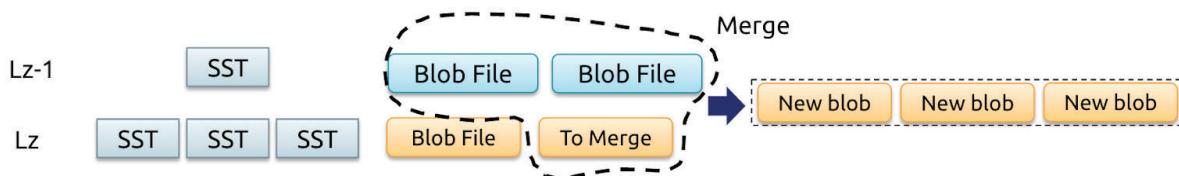


Figure 472: RangeMerge

Therefore, the Range Merge operation is needed to keep the number of sorted runs within a certain level. At the time of `OnCompactionComplete`, Titan counts the number of sorted runs in a range. If the number is large, Titan marks the corresponding blob file as `ToMerge` and rewrites it in the next compaction.

11.11.1.4 Titan Configuration

This document introduces how to enable and disable [Titan](#) using the corresponding configuration items, as well as the relevant parameters and the Level Merge feature.

11.11.1.4.1 Enable Titan

Titan is compatible with RocksDB, so you can directly enable Titan on the existing TiKV instances that use RocksDB. You can use one of the following two methods to enable Titan:

- Method 1: If you have deployed the cluster using TiUP, you can execute the `tiup cluster edit-config ${cluster-name}` command and edit the TiKV configuration file as the following example shows:

```
tikv:
  rocksdb.titan.enabled: true
```

Reload the configuration and TiKV will be rolling restarted online:

```
tiup cluster reload ${cluster-name} -R tikv
```

For the detailed command, see [Modify the configuration using TiUP](#).

- Method 2: Directly edit the TiKV configuration file to enable Titan (**NOT** recommended for the production environment).

```
[rocksdb.titan]
enabled = true
```

After Titan is enabled, the existing data stored in RocksDB is not immediately moved to the Titan engine. As new data is written to the TiKV foreground and RocksDB performs compaction, the values are progressively separated from keys and written to Titan. You can view the **TiKV Details -> Titan kv -> blob file size** panel to confirm the size of the data stored in Titan.

If you want to speed up the writing process, compact data of the whole TiKV cluster manually using tikv-ctl. For details, see [manual compaction](#).

Note:

When Titan is disabled, RocksDB cannot read data that has been migrated to Titan. If Titan is incorrectly disabled on a TiKV instance with Titan already enabled (mistakenly set `rocksdb.titan.enabled` to `false`), TiKV will fail to start, and the `You have disabled titan when its data directory is ↵ not empty` error appears in the TiKV log. To correctly disable Titan, see [Disable Titan](#).

11.11.1.4.2 Parameters

To adjust Titan-related parameters using TiUP, refer to [Modify the configuration](#).

- Titan GC thread count.

From the **TiKV Details -> Thread CPU -> RocksDB CPU** panel, if you observe that the Titan GC threads are at full capacity for a long time, consider increasing the size of the Titan GC thread pool.

```
[rocksdb.titan]
max-background-gc = 1
```

- Value size threshold.

When the size of the value written to the foreground is smaller than the threshold, this value is stored in RocksDB; otherwise, this value is stored in the blob file of Titan. Based on the distribution of value sizes, if you increase the threshold, more values are stored in RocksDB and TiKV performs better in reading small values. If you decrease the threshold, more values go to Titan, which further reduces RocksDB compactions.

```
[rocksdb.defaultcf.titan]
min-blob-size = "1KB"
```

- The algorithm used for compressing values in Titan, which takes value as the unit.

```
[rocksdb.defaultcf.titan]
blob-file-compression = "lz4"
```

- The size of value caches in Titan.

Larger cache size means higher read performance of Titan. However, too large a cache size causes Out of Memory (OOM). It is recommended to set the value of `storage.block-cache.capacity` to the store size minus the blob file size and set `blob.cache-size` to `memory size * 50% - block cache size` according to the monitoring metrics when the database is running stably. This maximizes the blob cache size when the block cache is large enough for the whole RocksDB engine.

```
[rocksdb.defaultcf.titan]
blob-cache-size = 0
```

- When the ratio of discardable data (the corresponding key has been updated or deleted) in a blob file exceeds the following threshold, Titan GC is triggered.

```
discardable-ratio = 0.5
```

When Titan writes the useful data of this blob file to another file, you can use the `discardable-ratio` value to estimate the upper limits of write amplification and space amplification (assuming the compression is disabled).

Upper limit of write amplification = $1 / \text{discardable_ratio}$

Upper limit of space amplification = $1 / (1 - \text{discardable_ratio})$

From the two equations above, you can see that decreasing the value of `discardable_ratio` can reduce space amplification but causes GC to be more frequent in Titan. Increasing the value reduces Titan GC, the corresponding I/O bandwidth, and CPU consumption but increases disk usage.

- The following option limits the I/O rate of RocksDB compaction. During peak traffic, limiting RocksDB compaction, its I/O bandwidth, and its CPU consumption reduces its impact on the write and read performance of the foreground.

When Titan is enabled, this option limits the summed I/O rates of RocksDB compaction and Titan GC. If you find that the I/O and/or CPU consumption of RocksDB compaction and Titan GC is too large, set this option to a suitable value according the disk I/O bandwidth and the actual write traffic.

```
[rocksdb]
rate-bytes-per-sec = 0
```

11.11.1.4.3 Disable Titan (experimental)

To disable Titan, you can configure the `rocksdb.defaultcf.titan.blob-run-mode` option. The optional values for `blob-run-mode` are as follows:

- When the option is set to `normal`, Titan performs read and write operations normally.
- When the option is set to `read-only`, all newly written values are written into RocksDB, regardless of the value size.
- When the option is set to `fallback`, all newly written values are written into RocksDB, regardless of the value size. Also, all compacted values stored in the Titan blob file are automatically moved back to RocksDB.

To disable Titan, set `blob-run-mode = "fallback"` and perform a full compaction using tikv-ctl. After that, check the monitoring metrics, confirm that the blob file size decreases to 0. Then you can set `rocksdb.titan.enabled` to `false` and restart TiKV.

Warning:

Disabling Titan is an experimental feature. It is **NOT** recommended to use it if not necessary.

11.11.1.4.4 Level Merge (experimental)

In TiKV 4.0, [Level Merge](#), a new algorithm, is introduced to improve the performance of range query and to reduce the impact of Titan GC on the foreground write operations. You can enable Level Merge using the following option:

```
[rocksdb.defaultcf.titan]  
level-merge = true
```

Enabling Level Merge has the following benefits:

- Greatly improve the performance of Titan range query.
- Reduce the impact of Titan GC on the foreground write operations and improve write performance.
- Reduce space amplification of Titan and the disk usage (compared to the disk usage with the default configuration).

Accordingly, the write amplification with Level Merge enabled is slightly higher than that of Titan but is still lower than that of the native RocksDB.

11.11.2 TiFlash

11.11.2.1 TiFlash Overview

TiFlash is the key component that makes TiDB essentially an Hybrid Transactional/-Analytical Processing (HTAP) database. As a columnar storage extension of TiKV, TiFlash provides both good isolation level and strong consistency guarantee.

In TiFlash, the columnar replicas are asynchronously replicated according to the Raft Learner consensus algorithm. When these replicas are read, the Snapshot Isolation level of consistency is achieved by validating Raft index and multi-version concurrency control (MVCC).

11.11.2.1.1 Architecture

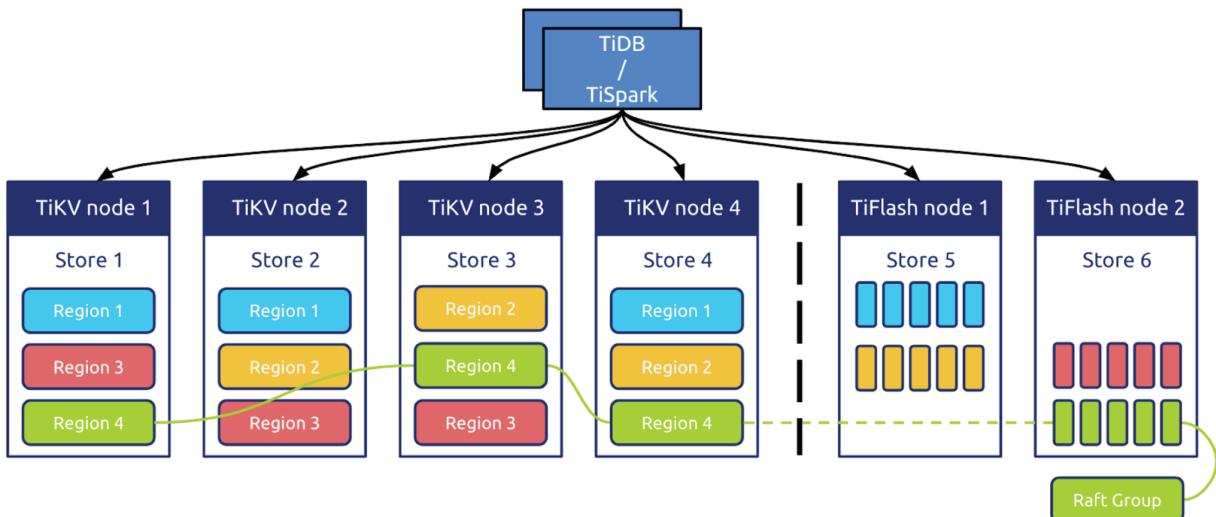


Figure 473: TiFlash Architecture

The above figure is the architecture of TiDB in its HTAP form, including TiFlash nodes.

TiFlash provides the columnar storage, with a layer of coprocessors efficiently implemented by ClickHouse. Similar to TiKV, TiFlash also has a Multi-Raft system, which supports replicating and distributing data in the unit of Region (see [Data Storage](#) for details).

TiFlash conducts real-time replication of data in the TiKV nodes at a low cost that does not block writes in TiKV. Meanwhile, it provides the same read consistency as in TiKV and ensures that the latest data is read. The Region replica in TiFlash is logically identical to those in TiKV, and is split and merged along with the Leader replica in TiKV at the same time.

TiFlash is compatible with both TiDB and TiSpark, which enables you to freely choose between these two computing engines.

It is recommended that you deploy TiFlash in different nodes from TiKV to ensure workload isolation. It is also acceptable to deploy TiFlash and TiKV in the same node if no business isolation is required.

Currently, data cannot be written directly into TiFlash. You need to write data in TiKV and then replicate it to TiFlash, because it connects to the TiDB cluster as a Learner role. TiFlash supports data replication in the unit of table, but no data is replicated by default after deployment. To replicate data of a specified table, see [Create TiFlash replicas for tables](#).

TiFlash has three components: the columnar storage module, `tiflash proxy`, and `pd buddy`. `tiflash proxy` is responsible for the communication using the Multi-Raft consensus algorithm. `pd buddy` works with PD to replicate data from TiKV to TiFlash in the unit of table.

When TiDB receives the DDL command to create replicas in TiFlash, the `pd buddy` component acquires the information of the table to be replicated via the status port of TiDB, and sends the information to PD. Then PD performs the corresponding data scheduling according to the information provided by `pd buddy`.

11.11.2.1.2 Key features

TiFlash has the following key features:

- Asynchronous replication
- Consistency
- Intelligent choice
- Computing acceleration

Asynchronous replication

The replica in TiFlash is asynchronously replicated as a special role, Raft Learner. This means when the TiFlash node is down or high network latency occurs, applications in TiKV can still proceed normally.

This replication mechanism inherits two advantages of TiKV: automatic load balancing and high availability.

- TiFlash does not rely on additional replication channels, but directly receives data from TiKV in a many-to-many manner.
- As long as the data is not lost in TiKV, you can restore the replica in TiFlash at any time.

Consistency

TiFlash provides the same Snapshot Isolation level of consistency as TiKV, and ensures that the latest data is read, which means that you can read the data previously written in TiKV. Such consistency is achieved by validating the data replication progress.

Every time TiFlash receives a read request, the Region replica sends a progress validation request (a lightweight RPC request) to the Leader replica. TiFlash performs the read operation only after the current replication progress includes the data covered by the timestamp of the read request.

Intelligent choice

TiDB can automatically choose to use TiFlash (column-wise) or TiKV (row-wise), or use both of them in one query to ensure the best performance.

This selection mechanism is similar to that of TiDB which chooses different indexes to execute query. TiDB optimizer makes the appropriate choice based on statistics of the read cost.

Computing acceleration

TiFlash accelerates the computing of TiDB in two ways:

- The columnar storage engine is more efficient in performing read operation.
- TiFlash shares part of the computing workload of TiDB.

TiFlash shares the computing workload in the same way as the TiKV Coprocessor does: TiDB pushes down the computing that can be completed in the storage layer. Whether the computing can be pushed down depends on the support of TiFlash. For details, see [Supported pushdown calculations](#).

11.11.2.1.3 See also

- To deploy a new cluster with TiFlash nodes, see [Deploy a TiDB cluster using TiUP](#).
- To add a TiFlash node in a deployed cluster, see [Scale out a TiFlash cluster](#).
- [Use TiFlash](#).
- [Maintain a TiFlash cluster](#).
- [Tune TiFlash performance](#).
- [Configure TiFlash](#).
- [Monitor the TiFlash cluster](#).
- Learn [TiFlash alert rules](#).
- [Troubleshoot a TiFlash cluster](#).

11.11.2.2 Use TiFlash

After TiFlash is deployed, data replication does not automatically begin. You need to manually specify the tables to be replicated.

You can either use TiDB to read TiFlash replicas for medium-scale analytical processing, or use TiSpark to read TiFlash replicas for large-scale analytical processing, which is based on your own needs. See the following sections for details:

- Use TiDB to read TiFlash replicas
- Use TiSpark to read TiFlash replicas

11.11.2.2.1 Create TiFlash replicas for tables

After TiFlash is connected to the TiKV cluster, data replication by default does not begin. You can send a DDL statement to TiDB through a MySQL client to create a TiFlash replica for a specific table:

```
ALTER TABLE table_name SET TIFLASH REPLICA count
```

The parameter of the above command is described as follows:

- **count** indicates the number of replicas. When the value is 0, the replica is deleted.

If you execute multiple DDL statements on a same table, only the last statement is ensured to take effect. In the following example, two DDL statements are executed on the table `tpch50`, but only the second statement (to delete the replica) takes effect.

Create two replicas for the table:

```
ALTER TABLE `tpch50`.`lineitem` SET TIFLASH REPLICA 2
```

Delete the replica:

```
ALTER TABLE `tpch50`.`lineitem` SET TIFLASH REPLICA 0
```

Notes:

- If the table `t` is replicated to TiFlash through the above DDL statements, the table created using the following statement will also be automatically replicated to TiFlash:

```
CREATE TABLE table_name like t
```

- For versions earlier than v4.0.6, if you create the TiFlash replica before using TiDB Lightning to import the data, the data import will fail. You must import data to the table before creating the TiFlash replica for the table.
- If TiDB and TiDB Lightning are both v4.0.6 or later, no matter a table has TiFlash replica(s) or not, you can import data to that table using TiDB Lightning. Note that this might slow the TiDB Lightning procedure, which depends on the NIC bandwidth on the lightning host, the CPU and disk load of the TiFlash node, and the number of TiFlash replicas.
- It is recommended that you do not replicate more than 1,000 tables because this lowers the PD scheduling performance. This limit will be removed in later versions.
- In v5.1 and later versions, setting the replicas for the system tables is no longer supported. Before upgrading the cluster, you need to clear the replicas of the relevant system tables. Otherwise, you cannot modify the replica settings of the system tables after you upgrade the cluster to a later version.

11.11.2.2.2 Check the replication progress

You can check the status of the TiFlash replicas of a specific table using the following statement. The table is specified using the `WHERE` clause. If you remove the `WHERE` clause, you will check the replica status of all tables.

```
SELECT * FROM information_schema.tiflash_replica WHERE TABLE_SCHEMA = '<
→ db_name>' AND TABLE_NAME = '<table_name>'
```

In the result of above statement:

- `AVAILABLE` indicates whether the TiFlash replicas of this table is available or not. 1 means available and 0 means unavailable. Once the replicas become available, this status does not change. If you use DDL statements to modify the number of replicas, the replication status will be recalculated.
- `PROGRESS` means the progress of the replication. The value is between 0.0 and 1.0. 1 means at least one replica is replicated.

11.11.2.2.3 Use TiDB to read TiFlash replicas

TiDB provides three ways to read TiFlash replicas. If you have added a TiFlash replica without any engine configuration, the CBO (cost-based optimization) mode is used by default.

Smart selection

For tables with TiFlash replicas, the TiDB optimizer automatically determines whether to use TiFlash replicas based on the cost estimation. You can use the `desc` or `explain → analyze` statement to check whether or not a TiFlash replica is selected. For example:

```
desc select count(*) from test.t;
```

id	estRows	task	access object	operator
StreamAgg_9	1.00	root		funcs:count(1)
->Column#4				
-TableReader_17	1.00	root		data:
-TableFullScan_16				
-TableFullScan_16	1.00	cop[tiflash]	table:t	keep order:
stats:pseudo				
3 rows in set (0.00 sec)				

```
explain analyze select count(*) from test.t;
```

id	estRows	actRows	task	access object	operator
info	memory	disk			
StreamAgg_9	1.00	1	root		time
↳ :83.8372ms, loops:2					funcs:count
↳ (1)->Column#4	372 Bytes	N/A			
TableReader_17	1.00	1	root		time
↳ :83.7776ms, loops:2, rpc num: 1, rpc time:83.5701ms, proc keys:0					
↳ data:TableFullScan_16	152 Bytes	N/A			
TableFullScan_16	1.00	1	cop[tiflash]	table:t	time
↳ :43ms, loops:1					keep order:
↳ false, stats:pseudo	N/A	N/A			

cop[tiflash] means that the task will be sent to TiFlash for processing. If you have not selected a TiFlash replica, you can try to update the statistics using the `analyze table` statement, and then check the result using the `explain analyze` statement.

Note that if a table has only a single TiFlash replica and the related node cannot provide service, queries in the CBO mode will repeatedly retry. In this situation, you need to specify the engine or use the manual hint to read data from the TiKV replica.

Engine isolation

Engine isolation is to specify that all queries use a replica of the specified engine by configuring the corresponding variable. The optional engines are “tikv”, “tidb” (indicates the internal memory table area of TiDB, which stores some TiDB system tables and cannot be actively used by users), and “tiflash”, with the following two configuration levels:

- TiDB instance-level, namely, INSTANCE level. Add the following configuration item in the TiDB configuration file:

```
[isolation-read]
engines = ["tikv", "tidb", "tiflash"]
```

The INSTANCE-level default configuration is `["tikv", "tidb", "tiflash"]`
 ↳ .

- SESSION level. Use the following statement to configure:

```
set @@session.tidb_isolation_read_engines = "engine list separated by
↪ commas";
```

or

```
set SESSION tidb_isolation_read_engines = "engine list separated by
↪ commas";
```

The default configuration of the SESSION level inherits from the configuration of the TiDB INSTANCE level.

The final engine configuration is the session-level configuration, that is, the session-level configuration overrides the instance-level configuration. For example, if you have configured “tikv” in the INSTANCE level and “tiflash” in the SESSION level, then the TiFlash replicas are read. If the final engine configuration is “tikv” and “tiflash”, then the TiKV and TiFlash replicas are both read, and the optimizer automatically selects a better engine to execute.

Note:

Because [TiDB Dashboard](#) and other components need to read some system tables stored in the TiDB memory table area, it is recommended to always add the “tidb” engine to the instance-level engine configuration.

If the queried table does not have a replica of the specified engine (for example, the engine is configured as “tiflash” but the table does not have a TiFlash replica), the query returns an error.

Manual hint

Manual hint can force TiDB to use specified replicas for specific table(s) on the premise of satisfying engine isolation. Here is an example of using the manual hint:

```
select /*+ read_from_storage(tiflash[table_name]) */ ... from table_name;
```

If you set an alias to a table in a query statement, you must use the alias in the statement that includes a hint for the hint to take effect. For example:

```
select /*+ read_from_storage(tiflash[alias_a,alias_b]) */ ... from
↪ table_name_1 as alias_a, table_name_2 as alias_b where alias_a.
↪ column_1 = alias_b.column_2;
```

In the above statements, `tiflash[]` prompts the optimizer to read the TiFlash replicas. You can also use `tikv[]` to prompt the optimizer to read the TiKV replicas as needed. For hint syntax details, refer to [READ_FROM_STORAGE](#).

If the table specified by a hint does not have a replica of the specified engine, the hint is ignored and a warning is reported. In addition, a hint only takes effect on the premise of engine isolation. If the engine specified in a hint is not in the engine isolation list, the hint is also ignored and a warning is reported.

Note:

The MySQL client of 5.7.7 or earlier versions clears optimizer hints by default. To use the hint syntax in these early versions, start the client with the → `--comments` option, for example, `mysql -h 127.0.0.1 -P 4000 -uroot --comments`.

The relationship of smart selection, engine isolation, and manual hint

In the above three ways of reading TiFlash replicas, engine isolation specifies the overall range of available replicas of engines; within this range, manual hint provides statement-level and table-level engine selection that is more fine-grained; finally, CBO makes the decision and selects a replica of an engine based on cost estimation within the specified engine list.

Note:

Before v4.0.3, the behavior of reading from TiFlash replica in a non-read-only SQL statement (for example, `INSERT INTO ... SELECT`, `SELECT ... FOR UPDATE`, `UPDATE ...`, `DELETE ...`) is undefined. In v4.0.3 and later versions, internally TiDB ignores the TiFlash replica for a non-read-only SQL statement to guarantee the data correctness. That is, for **smart selection**, TiDB automatically selects the non-TiFlash replica; for **engine isolation** that specifies TiFlash replica **only**, TiDB reports an error; and for **manual hint**, TiDB ignores the hint.

11.11.2.2.4 Use TiSpark to read TiFlash replicas

Currently, you can use TiSpark to read TiFlash replicas in a method similar to the engine isolation in TiDB. This method is to configure the `spark.tispark.isolation_read_engines` parameter. The parameter value defaults to `tikv,tiflash`, which means that TiDB reads data from TiFlash or from TiKV according to CBO's selection. If you set the parameter value to `tiflash`, it means that TiDB forcibly reads data from TiFlash.

Notes

When this parameter is set to `true`, only the TiFlash replicas of all tables involved in the query are read and these tables must have TiFlash replicas; for tables that do not have TiFlash replicas, an error is reported. When this parameter is set to `false`, only the TiKV replica is read.

You can configure this parameter in one of the following ways:

- Add the following item in the `spark-defaults.conf` file:

```
spark.tispark.isolation_read_engines tiflash
```

- Add `--conf spark.tispark.isolation_read_engines=tiflash` in the initialization command when initializing Spark shell or Thrift server.
- Set `spark.conf.set("spark.tispark.isolation_read_engines", "tiflash")` in Spark shell in a real-time manner.
- Set `set spark.tispark.isolation_read_engines=tiflash` in Thrift server after the server is connected via beeline.

11.11.2.2.5 Supported push-down calculations

TiFlash supports the push-down of the following operators:

- TableScan: Reads data from tables.
- Selection: Filters data.
- HashAgg: Performs data aggregation based on the [Hash Aggregation](#) algorithm.
- StreamAgg: Performs data aggregation based on the [Stream Aggregation](#) algorithm. SteamAgg only supports the aggregation without the `GROUP BY` condition.
- TopN: Performs the TopN calculation.
- Limit: Performs the limit calculation.
- Project: Performs the projection calculation.
- HashJoin (Equi Join): Performs the join calculation based on the [Hash Join](#) algorithm, but with the following conditions:
 - The operator can be pushed down only in the [MPP mode](#).
 - The push-down of `Full Outer Join` is not supported.
- HashJoin (Non-Equi Join): Performs the Cartesian Join algorithm, but with the following conditions:
 - The operator can be pushed down only in the [MPP mode](#).

- Cartesian Join is supported only in Broadcast Join.

In TiDB, operators are organized in a tree structure. For an operator to be pushed down to TiFlash, all of the following prerequisites must be met:

- All of its child operators can be pushed down to TiFlash.
- If an operator contains expressions (most of the operators contain expressions), all expressions of the operator can be pushed down to TiFlash.

Currently, TiFlash supports the following push-down expressions:

- Mathematical functions: +, -, /, *, %, >=, <=, =, !=, <, >, round, abs, ↪ floor(int), ceil(int), ceiling(int), sqrt, log, log2, log10, ln, exp ↪ , pow, sign, radians, degrees, conv, crc32
- Logical functions: and, or, not, case when, if, ifnull, isnull, in, like, ↪ coalesce
- Bitwise operations: bitand, bitor, bigneg, bitxor
- String functions: substr, char_length, replace, concat, concat_ws, left, ↪ right, ascii, length, trim, ltrim, rtrim, position, format, lower, ↪ ucase, upper, substring_index
- Date functions: date_format, timestampdiff, from_unixtime, unix_timestamp ↪ (int), unix_timestamp(decimal), str_to_date(date), str_to_date(↪ datetime), datediff, year, month, day, extract(datetime), date, ↪ hour, microsecond, minute, second, sysdate
- JSON function: json_length
- Conversion functions: cast(int as double), cast(int as decimal), cast(int ↪ as string), cast(int as time), cast(double as int), cast(double as ↪ decimal), cast(double as string), cast(double as time), cast(string ↪ as int), cast(string as double), cast(string as decimal), cast(↪ string as time), cast(decimal as int), cast(decimal as string), cast(↪ (decimal as time), cast(time as int), cast(time as decimal), cast(↪ time as string), cast(time as real)
- Aggregate functions: min, max, sum, count, avg, approx_count_distinct, ↪ group_concat
- Miscellaneous functions: inetntoa, inetaton, inet6ntoa, inet6aton

In addition, expressions that contain the Bit/Set/Geometry type cannot be pushed down to TiFlash.

If a query encounters unsupported push-down calculations, TiDB needs to complete the remaining calculations, which might greatly affect the TiFlash acceleration effect. The currently unsupported operators and expressions might be supported in future versions.

11.11.2.2.6 Use the MPP mode

TiFlash supports using the MPP mode to execute queries, which introduces cross-node data exchange (data shuffle process) into the computation. TiDB automatically determines whether to select the MPP mode using the optimizer's cost estimation. You can change the selection strategy by modifying the values of `tidb_allow_mpp` and `tidb_enforce_mpp`.

Control whether to select the MPP mode

The `tidb_allow_mpp` variable controls whether TiDB can select the MPP mode to execute queries. The `tidb_enforce_mpp` variable controls whether the optimizer's cost estimation is ignored and the MPP mode of TiFlash is forcibly used to execute queries.

The results corresponding to all values of these two variables are as follows:

	<code>tidb_allow_mpp=off</code>	<code>tidb_allow_mpp=on</code> (by default)
<code>tidb_enforce_mpp=off</code> (by default)	The MPP mode is not used.	The optimizer selects the MPP mode based on cost estimation. (by default)
<code>tidb_enforce_mpp=on</code>	The MPP mode is not used.	TiDB ignores the cost estimation and selects the MPP mode.

For example, if you do not want to use the MPP mode, you can execute the following statements:

```
set @@session.tidb_allow_mpp=1;
set @@session.tidb_enforce_mpp=0;
```

If you want TiDB's cost-based optimizer to automatically decide whether to use the MPP mode (by default), you can execute the following statements:

```
set @@session.tidb_allow_mpp=1;
set @@session.tidb_enforce_mpp=0;
```

If you want TiDB to ignore the optimizer's cost estimation and to forcibly select the MPP mode, you can execute the following statements:

```
set @@session.tidb_allow_mpp=1;
set @@session.tidb_enforce_mpp=1;
```

The initial value of the `tidb_enforce_mpp` session variable is equal to the `enforce-mpp` configuration value of this tidb-server instance (which is `false` by default). If multiple tidb-server instances in a TiDB cluster only perform analytical queries and you want to make sure that the MPP mode is used on these instances, you can change their `enforce-mpp` configuration values to `true`.

Note:

When `tidb_enforce_mpp=1` takes effect, the TiDB optimizer will ignore the cost estimation to choose the MPP mode. However, if other factors block the MPP mode, TiDB will not select the MPP mode. These factors include the absence of TiFlash replica, unfinished replication of TiFlash replicas, and statements containing operators or functions that are not supported by the MPP mode.

If TiDB optimizer cannot select the MPP mode due to reasons other than cost estimation, when you use the `EXPLAIN` statement to check out the execution plan, a warning is returned to explain the reason. For example:

```
set @@session.tidb_enforce_mpp=1;
create table t(a int);
explain select count(*) from t;
show warnings;
```

Level	Code	Message
Warning	1105	MPP mode may be blocked because there aren't tiflash replicas of table `t`.

Algorithm support for the MPP mode

The MPP mode supports these physical algorithms: Broadcast Hash Join, Shuffled Hash Join, Shuffled Hash Aggregation, Union All, TopN, and Limit. The optimizer automatically determines which algorithm to be used in a query. To check the specific query execution plan, you can execute the `EXPLAIN` statement. If the result of the `EXPLAIN` statement shows `ExchangeSender` and `ExchangeReceiver` operators, it indicates that the MPP mode has taken effect.

The following statement takes the table structure in the TPC-H test set as an example:

```
explain select count(*) from customer c join nation n on c.c_nationkey=n.
    ↵ n_nationkey;
```

```

+--+
| id          | estRows | task      | access
| object | operator info
|       |
+--+
| HashAgg_23           | 1.00    | root     |
|   funcs:count(Column#16)->Column#15
|   |
|   -TableReader_25      | 1.00    | root     |
|     data:ExchangeSender_24
|     |
|     - ExchangeSender_24    | 1.00    | batchCop[tiflash] |
|       ExchangeType: PassThrough
|       |
|     - HashAgg_12          | 1.00    | batchCop[tiflash] |
|       funcs:count(1)->Column#16
|       |
|     - HashJoin_17         | 3000000.00 | batchCop[tiflash] |
|       inner join, equal:[eq(tpch.nation.n_nationkey, tpch.
|       customer.c_nationkey)] |
|       - ExchangeReceiver_21(Build) | 25.00    | batchCop[tiflash] |
|         |
|         - ExchangeSender_20      | 25.00    | batchCop[tiflash] |
|           ExchangeType: Broadcast
|           |
|           - TableFullScan_18      | 25.00    | batchCop[tiflash] |
|             table:n | keep order:false
|             |
|             - TableFullScan_22(Probe) | 3000000.00 | batchCop[tiflash] |
|               table:c | keep order:false
|               |
+--+
|       |
|       |
9 rows in set (0.00 sec)

```

In the example execution plan, the `ExchangeReceiver` and `ExchangeSender` operators are included. The execution plan indicates that after the `nation` table is read, the `ExchangeSender` operator broadcasts the table to each node, the `HashJoin` and `HashAgg` operations are performed on the `nation` table and the `customer` table, and then the results

are returned to TiDB.

TiFlash provides the following two global/session variables to control whether to use Broadcast Hash Join:

- **`tidb_broadcast_join_threshold_size`**: The unit of the value is bytes. If the table size (in the unit of bytes) is less than the value of the variable, the Broadcast Hash Join algorithm is used. Otherwise, the Shuffled Hash Join algorithm is used.
- **`tidb_broadcast_join_threshold_count`**: The unit of the value is rows. If the objects of the join operation belong to a subquery, the optimizer cannot estimate the size of the subquery result set, so the size is determined by the number of rows in the result set. If the estimated number of rows in the subquery is less than the value of this variable, the Broadcast Hash Join algorithm is used. Otherwise, the Shuffled Hash Join algorithm is used.

11.11.2.2.7 Notes

Currently, TiFlash does not support some features. These features might be incompatible with the native TiDB:

- In the TiFlash computation layer:
 - Checking overflowed numerical values is not supported. For example, adding two maximum values of the BIGINT type `9223372036854775807 + ↪ 9223372036854775807`. The expected behavior of this calculation in TiDB is to return the `ERROR 1690 (22003): BIGINT value is out of range` error. However, if this calculation is performed in TiFlash, an overflow value of `-2` is returned without any error.
 - The window function is not supported.
 - Reading data from TiKV is not supported.
 - Currently, the `sum` function in TiFlash does not support the string-type argument. But TiDB cannot identify whether any string-type argument has been passed into the `sum` function during the compiling. Therefore, when you execute statements similar to `select sum(string_col)from t`, TiFlash returns the `[FLASH:Coprocessor:Unimplemented] CastStringAsReal is not supported ↪ .` error. To avoid such an error in this case, you need to modify this SQL statement to `select sum(cast(string_col as double))from t`.
 - Currently, TiFlash's decimal division calculation is incompatible with that of TiDB. For example, when dividing decimal, TiFlash performs the calculation always using the type inferred from the compiling. However, TiDB performs this calculation using a type that is more precise than that inferred from the compiling. Therefore, some SQL statements involving the decimal division return different execution results when executed in TiDB + TiKV and in TiDB + TiFlash. For example:

```

mysql> create table t (a decimal(3,0), b decimal(10, 0));
Query OK, 0 rows affected (0.07 sec)
mysql> insert into t values (43, 1044774912);
Query OK, 1 row affected (0.03 sec)
mysql> alter table t settiflash replica 1;
Query OK, 0 rows affected (0.07 sec)
mysql> set session tidb_isolation_read_engines='tikv';
Query OK, 0 rows affected (0.00 sec)
mysql> select a/b, a/b + 0.000000000001 from t where a/b;
+-----+-----+
| a/b   | a/b + 0.000000000001 |
+-----+-----+
| 0.0000 |      0.0000000410001 |
+-----+-----+
1 row in set (0.00 sec)
mysql> set session tidb_isolation_read_engines='tiflash';
Query OK, 0 rows affected (0.00 sec)
mysql> select a/b, a/b + 0.000000000001 from t where a/b;
Empty set (0.01 sec)

```

In the example above, `a/b`'s inferred type from the compiling is `Decimal(7,4)` both in TiDB and in TiFlash. Constrained by `Decimal(7,4)`, `a/b`'s returned type should be `0.0000`. In TiDB, `a/b`'s runtime precision is higher than `Decimal(7,4)`, so the original table data is not filtered by the `where a/b` condition. However, in TiFlash, the calculation of `a/b` uses `Decimal(7,4)` as the result type, so the original table data is filtered by the `where a/b` condition.

11.12 Telemetry

By default, TiDB, TiUP and TiDB Dashboard collect usage information and share the information with PingCAP to help understand how to improve the product. For example, this usage information helps prioritize new features.

11.12.1 What is shared?

The following sections describe the shared usage information in detail for each component. The usage details that get shared might change over time. These changes (if any) will be announced in [release notes](#).

Note:

In **ALL** cases, user data stored in the TiDB cluster will **NOT** be shared.
You can also refer to [PingCAP Privacy Policy](#).

11.12.1.1 TiDB

When the telemetry collection feature is enabled in TiDB, the TiDB cluster collects usage details on a 6-hour basis. These usage details include but are not limited to:

- A randomly generated telemetry ID.
- Deployment characteristics, such as the size of hardware (CPU, memory, disk), TiDB components versions, OS name.
- The status of query requests in the system, such as the number of query requests and the duration.
- Component usage, for example, whether the Async Commit feature is in use or not.

To view the full content of the usage information shared to PingCAP, execute the following SQL statement:

```
ADMIN SHOW TELEMETRY;
```

11.12.1.2 TiDB Dashboard

When the telemetry collection feature is enabled for TiDB Dashboard, usage information on the TiDB Dashboard web UI will be shared, including (but not limited to):

- A randomly generated telemetry ID.
- User operation information, such as the name of the TiDB Dashboard web page accessed by the user.
- Browser and OS information, such as browser name, OS name, and screen resolution.

To view the full content of the usage information shared to PingCAP, use the [Network Activity Inspector of Chrome DevTools](#) or the [Network Monitor of Firefox Developer Tools](#).

11.12.1.3 TiUP

When the telemetry collection feature is enabled in TiUP, user operations with TiUP will be shared, including (but not limited to):

- A randomly generated telemetry ID.
- Execution status of TiUP commands, such as whether the execution is successful and the execution duration.
- Deployment characteristics, such as the size of hardware, TiDB components versions, and deployment configuration names that have been modified.

To view the full content of the usage information shared to PingCAP, set the `TIUP_CLUSTER_DEBUG=enable` environment variable when executing the TiUP command. For example:

```
TIUP_CLUSTER_DEBUG=enable tiup cluster list
```

11.12.2 Disable telemetry

11.12.2.1 Disable TiDB telemetry at deployment

When deploying TiDB clusters, configure `enable-telemetry = false` to disable the TiDB telemetry collection on all TiDB instances. You can also use this setting to disable telemetry in an existing TiDB cluster, which does not take effect until you restart the cluster.

Detailed steps to disable telemetry in different deployment tools are listed below.

Binary deployment

Create a configuration file `tidb_config.toml` with the following content:

```
enable-telemetry = false
```

Specify the `--config=tidb_config.toml` command-line parameter when starting TiDB for the configuration file above to take effect.

See [TiDB Configuration Options](#) and [TiDB Configuration File](#) for details.

Deployment using TiUP Playground

Create a configuration file `tidb_config.toml` with the following content:

```
enable-telemetry = false
```

When starting TiUP Playground, specify the `--db.config tidb_config.toml` command-line parameter for the configuration file above to take effect. For example:

```
tiup playground --db.config tidb_config.toml
```

See [Quickly Deploy a Local TiDB Cluster](#) for details.

Deployment using TiUP Cluster

Modify the deployment topology file `topology.yaml` to add the following content:

```
server_configs:
  tidb:
    enable-telemetry: false
```

Deployment in Kubernetes via TiDB Operator

Configure `spec.tidb.config.enable-telemetry: false` in `tidb-cluster.yaml` or `TidbCluster Custom Resource`.

See [Deploy TiDB Operator in Kubernetes](#) for details.

Note:

This configuration item requires TiDB Operator v1.1.3 or later to take effect.

11.12.2.2 Disable TiDB telemetry for deployed TiDB clusters

In existing TiDB clusters, you can also modify the system variable `tidb_enable_telemetry` ↵ to dynamically disable the TiDB telemetry collection:

```
SET GLOBAL tidb_enable_telemetry = 0;
```

Note:

When you disable telemetry, the configuration file has a higher priority over system variable. That is, after telemetry collection is disabled by the configuration file, the value of the system variable will be ignored.

11.12.2.3 Disable TiDB Dashboard telemetry

Configure `dashboard.enable-telemetry = false` to disable the TiDB Dashboard telemetry collection on all PD instances. You need to restart the running clusters for the configuration to take effect.

Detailed steps to disable telemetry for different deployment tools are listed below.

Binary deployment

Create a configuration file `pd_config.toml` with the following content:

```
[dashboard]
enable-telemetry = false
```

Specify the `--config=pd_config.toml` command-line parameter when starting PD to take effect.

See [PD Configuration Flags](#) and [PD Configuration File](#) for details.

Deployment using TiUP Playground

Create a configuration file `pd_config.toml` with the following content:

```
[dashboard]
enable-telemetry = false
```

When starting TiUP Playground, specify the `--pd.config pd_config.toml` command-line parameter to take effect, for example:

```
tiup playground --pd.config pd_config.toml
```

See [Quickly Deploy a Local TiDB Cluster](#) for details.

Deployment using TiUP Cluster

Modify the deployment topology file `topology.yaml` to add the following content:

```
server_configs:  
  pd:  
    dashboard.enable-telemetry: false
```

Deployment in Kubernetes via TiDB Operator

Configure `spec.pd.config.dashboard.enable-telemetry: false` in `tidb-cluster.yaml` or TidbCluster Custom Resource.

See [Deploy TiDB Operator in Kubernetes](#) for details.

Note:

This configuration item requires TiDB Operator v1.1.3 or later to take effect.

11.12.2.4 Disable TiUP telemetry

To disable the TiUP telemetry collection, execute the following command:

```
tiup telemetry disable
```

11.12.3 Check telemetry status

For TiDB telemetry, execute the following SQL statement to check the telemetry status:

```
ADMIN SHOW TELEMETRY;
```

If the `DATA_PREVIEW` column in the execution result is empty, TiDB telemetry is disabled. If not, TiDB telemetry is enabled. You can also check when the usage information was shared previously according to the `LAST_STATUS` column and whether the sharing was successful or not.

For TiUP telemetry, execute the following command to check the telemetry status:

```
tiup telemetry status
```

11.12.4 Compliance

To meet compliance requirements in different countries or regions, the usage information is sent to servers located in different countries according to the IP address of the sender machine:

- For IP addresses from the Chinese mainland, usage information is sent to and stored on cloud servers in the Chinese mainland.
- For IP addresses from outside of the Chinese mainland, usage information is sent to and stored on cloud servers in the US.

See [PingCAP Privacy Policy](#) for details.

11.13 Error Codes and Troubleshooting

This document describes the problems encountered during the use of TiDB and provides the solutions.

11.13.1 Error codes

TiDB is compatible with the error codes in MySQL, and in most cases returns the same error code as MySQL. For a list of error codes for MySQL, see [MySQL 5.7 Error Message Reference](#). In addition, TiDB has the following unique error codes:

Note:

Some error codes stand for internal errors. Normally, TiDB handles the error rather than return it to the user, so some error codes are not listed here.

If you encounter an error code that is not listed here, [contact PingCAP](#) for support.

- Error Number: 8001

The memory used by the request exceeds the threshold limit for the TiDB memory usage.

Increase the memory limit for a single SQL statement by configuring `mem-quota-query`.

- Error Number: 8002

To guarantee consistency, a transaction with the `SELECT FOR UPDATE` statement cannot be retried when it encounters a commit conflict. TiDB rolls back the transaction and returns this error.

The application can safely retry the whole transaction.

- Error Number: 8003

If the data in a row is not consistent with the index when executing the `ADMIN CHECK ↞ TABLE` command, TiDB returns this error. This error is commonly seen when you check the data corruption in the table.

You can [contact PingCAP](#) for support.

- Error Number: 8004

A single transaction is too large.

See [the error message `transaction too large`](#) for the cause and solution.

- Error Number: 8005

Transactions in TiDB encounter write conflicts.

See [the Troubleshoot section](#) for the cause and solution.

- Error Number: 8018

When you reload a plugin, if the plugin has not been loaded before, this error is returned.

You can execute an initial load of the plugin.

- Error Number: 8019

The version of the plugin that is being reloaded is different from the previous version. Therefore, the plugin cannot be reloaded, and this error is returned.

You can reload the plugin by ensuring that the plugin version is the same as the previous one.

- Error Number: 8020

When the table is locked, if you perform a write operation on the table, this error is returned.

Unlock the table and retry the write operation.

- Error Number: 8021

When the key to be read from TiKV does not exist, this error is returned. This error is used internally, and the external result is an empty read.

- Error Number: 8022

The transaction commit fails and has been rolled back.

The application can safely retry the whole transaction.

- Error Number: 8023

If you set an empty value when writing the transaction cache, this error is returned. This error is used and dealt with internally, and is not returned to the application.

- Error Number: 8024

Invalid transactions. If TiDB finds that no transaction ID (Start Timestamp) is obtained for the transaction that is being executed, which means this transaction is invalid, this error is returned.

Usually this error does not occur. If you encounter this error, [contact PingCAP](#) for support.

- Error Number: 8025

The single Key-Value pair being written is too large. The largest single Key-Value pair supported in TiDB is 6 MB by default.

If a pair exceeds this limit, you need to properly adjust the `txn-entry-size-limit` configuration value to relax the limit.

- Error Number: 8026

The interface function being used has not been implemented. This error is only used internally, and is not returned to the application.

- Error Number: 8027

The table schema version is outdated. TiDB applies schema changes online. When the table schema version of the TiDB server is earlier than that of the entire system, this error is returned if you execute a SQL statement.

When this error occurs, check the network between the TiDB server and the PD Leader.

- Error Number: 8028

TiDB does not support table lock, which is called metadata lock in MySQL and might be called intention lock in other databases.

When a transaction is executed, the transaction cannot recognize the table schema changes. Therefore, when committing a transaction, TiDB checks the table schema related the transaction. If the related table schema has changed during the execution, the transaction commit will fail and this error is returned.

The application can safely retry the whole transaction.

- Error Number: 8029

This error occurs when numeric conversion within the database encounters an error. This error is only used internally and is converted to a specific type of error for external applications.

- Error Number: 8030

After an unsigned positive integer is converted to a signed integer, it exceeds the maximum value and displays as a negative integer. This error mostly occurs in the alert message.

- Error Number: 8031

When being converted to an unsigned integer, a negative integer is converted to a positive integer. This error mostly occurs in the alert message.

- Error Number: 8032

Invalid `year` format is used. `year` only accepts 1, 2 or 4 digits.

- Error Number: 8033

Invalid `year` value is used. The valid range of `year` is (1901, 2155).

- Error Number: 8037

Invalid `mode` format is used in the `week` function. `mode` must be 1 digit within [0, 7].

- Error Number: 8038

The field fails to obtain the default value. This error is usually used internally, and is converted to a specific type of error for external applications.

- Error Number: 8040

Unsupported operations are performed. For example, you perform a table locking operation on a view or a sequence.

- Error Number: 8047

The value of the system variable is not supported. This error usually occurs in the alarm information when the user sets a variable value that is not supported in the database.

- Error Number: 8048

An unsupported database isolation level is set.

If you cannot modify the codes because you are using a third-party tool or framework, consider using `tidb_skip_isolation_level_check` to bypass this check.

```
set @@tidb_skip_isolation_level_check = 1;
```

- Error Number: 8050

An unsupported privilege type is set.

See [Privileges required for TiDB operations](#) for the solution.

- Error Number: 8051

Unknown data type is encountered when TiDB parses the Exec argument list sent by the client.

If you encounter this error, check the client. If the client is normal, [contact PingCAP](#) for support.

- Error Number: 8052

The serial number of the data packet from the client is incorrect.

If you encounter this error, check the client. If the client is normal, [contact PingCAP](#) for support.

- Error Number: 8055

The current snapshot is too old. The data may have been garbage collected. You can increase the value of `tidb_gc_life_time` to avoid this problem. TiDB automatically reserves data for long-running transactions. Usually this error does not occur.

See [garbage collection overview](#) and [garbage collection configuration](#).

- Error Number: 8059

The auto-random ID is exhausted and cannot be allocated. There is no way to recover from such errors currently. It is recommended to use bigint when using the auto random feature to obtain the maximum number of assignment. And try to avoid manually assigning values to the auto random column.

See [auto random](#) for reference.

- Error Number: 8060

Invalid auto-incrementing offset. Check the values of `auto_increment_increment` and `auto_increment_offset`.

- Error Number: 8061

Unsupported SQL Hint.

See [Optimizer Hints](#) to check and modify the SQL Hint.

- Error Number: 8062

An invalid token is used in SQL Hint. It conflicts with reserved words in SQL Hint.

See [Optimizer Hints](#) to check and modify the SQL Hint.

- Error Number: 8063

The limited memory usage set in SQL Hint exceeds the upper limit of the system. The setting in SQL Hint is ignored.

See [Optimizer Hints](#) to check and modify the SQL Hint.

- Error Number: 8064

It fails to parse SQL Hint.

See [Optimizer Hints](#) to check and modify the SQL Hint.

- Error Number: 8065

An invalid integer is used in SQL Hint.

See [Optimizer Hints](#) to check and modify the SQL Hint.

- Error Number: 8066

The second parameter in the `JSON_OBJECTAGG` function is invalid.

- Error Number: 8101

The format of plugin ID is incorrect.

The correct format is `[name]-[version]`, and no `-` is allowed in `name` and `version`.

- Error Number: 8102

Unable to read the plugin definition information.

Check the configuration related to the plugin.

- Error Number: 8103

The plugin name is incorrect.

Check the configuration of the plugin.

- Error Number: 8104

The plugin version does not match.

Check the configuration of the plugin.

- Error Number: 8105

The plugin is repeatedly loaded.

- Error Number: 8106

The plugin defines a system variable whose name does not begin with the plugin name.

Contact the developer of the plugin to modify.

- Error Number: 8107

The loaded plugin does not specify a version, or the specified version is too low.

Check the configuration of the plugin.

- Error Number: 8108

Unsupported execution plan type. This error is an internal error.

If you encounter this error, [contact PingCAP](#) for support.

- Error Number: 8109

The specified index cannot be found when the index is analyzed.

- Error Number: 8110

The Cartesian product operation cannot be executed.

Set `cross-join` in the configuration to `true`.

- Error Number: 8111

When executing the `EXECUTE` statement, the corresponding `Prepare` statement cannot be found.

- Error Number: 8112

The number of parameters in the `EXECUTE` statement is not consistent with the `Prepare` statement.

- Error Number: 8113

The table schema related in the `EXECUTE` statement has changed after the `Prepare` statement is executed.

- Error Number: 8115

It is not supported to prepare multiple lines of statements.

- Error Number: 8116

It is not supported to prepare DDL statements.

- Error Number: 8120

The `start tso` of transactions cannot be obtained.

Check the state/monitor/log of the PD server and the network between the TiDB server and the PD server.

- Error Number: 8121

Privilege check fails.

Check the privilege configuration of the database.

- Error Number: 8122

No corresponding table name is found, given the specified wild cards.

- Error Number: 8123

An SQL query with aggregate functions returns non-aggregated columns, which violates the `only_full_group_by` mode.

Modify the SQL statement or disable the `only_full_group_by` mode.

- Error Number: 8129

TiDB does not yet support JSON objects with the key length ≥ 65536 .

- Error Number: 8200

The DDL syntax is not yet supported.

See [compatibility of MySQL DDL](#) for reference.

- Error Number: 8214

The DDL operation is terminated by the `admin cancel` operation.

- Error Number: 8215

`ADMIN REPAIR TABLE` fails.

If you encounter this error, [contact PingCAP](#) for support.

- Error Number: 8216

The usage of automatic random columns is incorrect.

See [auto random](#) to modify.

- Error Number: 8223

This error occurs when detecting that the data is not consistent with the index.

If you encounter this error, [contact PingCAP](#) for support.

- Error Number: 8224

The DDL job cannot be found.

Check whether the job id specified by the `restore` operation exists.

- Error Number: 8225

The DDL operation is completed and cannot be canceled.

- Error Number: 8226

The DDL operation is almost completed and cannot be canceled.

- Error Number: 8227

Unsupported options are used when creating Sequence.

See [Sequence documentation](#) to find the list of the supported options.

- Error Number: 8228

Unsupported types are specified when using `setval` on Sequence.

See [Sequence documentation](#) to find the example of the function.

- Error Number: 8229

The transaction exceeds the survival time.

Commit or roll back the current transaction, and start a new transaction.

- Error Number: 8230

TiDB currently does not support using Sequence as the default value on newly added columns, and reports this error if you use it.

- Error Number: 9001

The PD request timed out.

Check the state/monitor/log of the PD server and the network between the TiDB server and the PD server.

- Error Number: 9002

The TiKV request timed out.

Check the state/monitor/log of the TiKV server and the network between the TiDB server and the TiKV server.

- Error Number: 9003

The TiKV server is busy and this usually occurs when the workload is too high.

Check the state/monitor/log of the TiKV server.

- Error Number: 9004

This error occurs when a large number of transactional conflicts exist in the database.

Check the code of application.

- Error Number: 9005

A certain Raft Group is not available, such as the number of replicas is not enough. This error usually occurs when the TiKV server is busy or the TiKV node is down.

Check the state/monitor/log of the TiKV server.

- Error Number: 9006

The interval of GC Life Time is too short and the data that should be read by the long transactions might be cleared.

Extend the interval of GC Life Time.

- Error Number: 9500

A single transaction is too large.

See [the error message transaction too large](#) for the solution.

- Error Number: 9007

Transactions in TiKV encounter write conflicts.

See [the Troubleshoot section](#) for the cause and solution.

- Error Number: 9008

Too many requests are sent to TiKV at the same time. The number exceeds limit.

Increase `tidb_store_limit` or set it to 0 to remove the limit on the traffic of requests.

- Error Number: 9010

TiKV cannot process this raft log.

Check the state/monitor/log of the TiKV server.

- Error Number: 9012

The TiFlash request timed out.

Check the state/monitor/log of the TiFlash server and the network between the TiDB server and TiFlash server.

- Error Number: 9013

The TiFlash server is busy and this usually occurs when the workload is too high.

Check the state/monitor/log of the TiFlash server.

11.13.2 Troubleshooting

See the [troubleshooting](#) and [FAQ](#) documents.

11.14 Table Filter

The TiDB ecosystem tools operate on all the databases by default, but oftentimes only a subset is needed. For example, you only want to work with the schemas in the form of `foo*` and `bar*` and nothing else.

Since TiDB 4.0, all TiDB ecosystem tools share a common filter syntax to define subsets. This document describes how to use the table filter feature.

11.14.1 Usage

11.14.1.1 CLI

Table filters can be applied to the tools using multiple `-f` or `--filter` command line parameters. Each filter is in the form of `db.table`, where each part can be a wildcard (further explained in the [next section](#)). The following lists the example usage in each tool.

- BR:

```
./br backup full -f 'foo.*' -f 'bar.*' -s 'local:///tmp/backup'
#                                     ^~~~~~
./br restore full -f 'foo.*' -f 'bar.*' -s 'local:///tmp/backup'
#                                     ^~~~~~
```

- Dumpling:

```
./dumpling -f 'foo.*' -f 'bar.*' -P 3306 -o /tmp/data/
#           ^~~~~~
```

- TiDB Lightning:

```
./tidb-lightning -f 'foo.*' -f 'bar.*' -d /tmp/data/ --backend tidb
#           ^~~~~~
```

11.14.1.2 TOML configuration files

Table filters in TOML files are specified as [array of strings](#). The following lists the example usage in each tool.

- TiDB Lightning:

```
[mydumper]
filter = ['foo.*', 'bar.*']
```

- TiCDC:

```
[filter]
rules = ['foo.*', 'bar.*']

[[sink.dispatchers]]
matcher = ['db1.*', 'db2.*', 'db3.*']
dispatcher = 'ts'
```

11.14.2 Syntax

11.14.2.1 Plain table names

Each table filter rule consists of a “schema pattern” and a “table pattern”, separated by a dot (.). Tables whose fully-qualified name matches the rules are accepted.

```
db1.tb11
db2.tb12
db3.tb13
```

A plain name must only consist of valid [identifier characters](#), such as:

- digits (0 to 9)
- letters (a to z, A to Z)
- \$
- -

- non ASCII characters (U+0080 to U+10FFFF)

All other ASCII characters are reserved. Some punctuations have special meanings, as described in the next section.

11.14.2.2 Wildcards

Each part of the name can be a wildcard symbol described in [fnmatch\(3\)](#):

- * — matches zero or more characters
- ? — matches one character
- [a-z] — matches one character between “a” and “z” inclusively
- [!a-z] — matches one character except “a” to “z”.

```
db[0-9].tbl[0-9a-f][0-9a-f]
data.*
*.backup_*
```

“Character” here means a Unicode code point, such as:

- U+00E9 (é) is 1 character.
- U+0065 U+0301 (é) are 2 characters.
- U+1F926 U+1F3FF U+200D U+2640 U+FE0F () are 5 characters.

11.14.2.3 File import

To import a file as the filter rule, include an @ at the beginning of the rule to specify the file name. The table filter parser treats each line of the imported file as additional filter rules.

For example, if a file config/filter.txt has the following content:

```
employees.*
*.WorkOrder
```

the following two invocations are equivalent:

```
./dumpling -f '@config/filter.txt'
./dumpling -f 'employees.*' -f '*.WorkOrder'
```

A filter file cannot further import another file.

11.14.2.4 Comments and blank lines

Inside a filter file, leading and trailing white-spaces of every line are trimmed. Furthermore, blank lines (empty strings) are ignored.

A leading # marks a comment and is ignored. # not at start of line is considered syntax error.

```
## this line is a comment
db.table # but this part is not comment and may cause error
```

11.14.2.5 Exclusion

An ! at the beginning of the rule means the pattern after it is used to exclude tables from being processed. This effectively turns the filter into a block list.

```
*.*
#^ note: must add the *.* to include all tables first
!*>Password
!employees.salaries
```

11.14.2.6 Escape character

To turn a special character into an identifier character, precede it with a backslash \.

```
db\.with\.dots.*
```

For simplicity and future compatibility, the following sequences are prohibited:

- \ at the end of the line after trimming whitespaces (use [] to match a literal whitespace at the end).
- \ followed by any ASCII alphanumeric character ([0-9a-zA-Z]). In particular, C-like escape sequences like \0, \r, \n and \t currently are meaningless.

11.14.2.7 Quoted identifier

Besides \, special characters can also be suppressed by quoting using " or `.

```
"db.with.dots"."tbl\1"
`db.with.dots`.`tbl\2`
```

The quotation mark can be included within an identifier by doubling itself.

```
"foo""bar".`foo``bar`
## equivalent to:
foo\"bar.foo`\bar
```

Quoted identifiers cannot span multiple lines.

It is invalid to partially quote an identifier:

```
"this is "invalid*.*
```

11.14.2.8 Regular expression

In case very complex rules are needed, each pattern can be written as a regular expression delimited with /:

```
/^db\d{2,}$/ ./^tbl\d{2,}$/
```

These regular expressions use the [Go dialect](#). The pattern is matched if the identifier contains a substring matching the regular expression. For instance, /b/ matches db01.

Note:

Every / in the regular expression must be escaped as \/, including inside [...] . You cannot place an unescaped / between \Q...\\E.

11.14.3 Multiple rules

When a table name matches none of the rules in the filter list, the default behavior is to ignore such unmatched tables.

To build a block list, an explicit *.* must be used as the first rule, otherwise all tables will be excluded.

```
## every table will be filtered out
./dumpling -f '!*.Password'

## only the "Password" table is filtered out, the rest are included.
./dumpling -f '*.*' -f '!*.Password'
```

In a filter list, if a table name matches multiple patterns, the last match decides the outcome. For instance:

```
## rule 1
employees.*
## rule 2
!* .dep*
## rule 3
*.departments
```

The filtered outcome is as follows:

Table name	Rule 1	Rule 2	Rule 3	Outcome
irrelevant.table				Default (reject)
employees.employees				Rule 1 (accept)
employees.dept_emp				Rule 2 (reject)
employees.departments				Rule 3 (accept)
else.departments				Rule 3 (accept)

Note:

In TiDB tools, the system schemas are always excluded in the default configuration. The system schemas are:

- INFORMATION_SCHEMA
- PERFORMANCE_SCHEMA
- METRICS_SCHEMA
- INSPECTION_SCHEMA
- mysql
- sys

11.15 Schedule Replicas by Topology Labels

Note:

TiDB v5.3.0 introduces an experimental support for [Placement Rules in SQL](#). This offers a more convenient way to configure the placement of tables and partitions. Placement Rules in SQL might replace placement configuration with PD in future releases.

To improve the high availability and disaster recovery capability of TiDB clusters, it is recommended that TiKV nodes are physically scattered as much as possible. For example, TiKV nodes can be distributed on different racks or even in different data centers. According to the topology information of TiKV, the PD scheduler automatically performs scheduling at the background to isolate each replica of a Region as much as possible, which maximizes the capability of disaster recovery.

To make this mechanism effective, you need to properly configure TiKV and PD so that the topology information of the cluster, especially the TiKV location information, is reported to PD during deployment. Before you begin, see [Deploy TiDB Using TiUP](#) first.

11.15.1 Configure labels based on the cluster topology

11.15.1.1 Configure labels for TiKV

You can use the command-line flag or set the TiKV configuration file to bind some attributes in the form of key-value pairs. These attributes are called **labels**. After TiKV is started, it reports its **labels** to PD so users can identify the location of TiKV nodes.

Assume that the topology has three layers: zone > rack > host, and you can use these labels (zone, rack, host) to set the TiKV location in one of the following methods:

- Use the command-line flag:

```
tikv-server --labels zone=<zone>,rack=<rack>,host=<host>
```

- Configure in the TiKV configuration file:

```
[server]
labels = "zone=<zone>,rack=<rack>,host=<host>"
```

11.15.1.2 Configure location-labels for PD

According to the description above, the label can be any key-value pair used to describe TiKV attributes. But PD cannot identify the location-related labels and the layer relationship of these labels. Therefore, you need to make the following configuration for PD to understand the TiKV node topology.

- If the PD cluster is not initialized, configure **location-labels** in the PD configuration file:

```
[replication]
location-labels = ["zone", "rack", "host"]
```

- If the PD cluster is already initialized, use the pd-ctl tool to make online changes:

```
pd-ctl config set location-labels zone,rack,host
```

The **location-labels** configuration is an array of strings, and each item corresponds to the key of TiKV **labels**. The sequence of each key represents the layer relationship of different labels.

Note:

You must configure **location-labels** for PD and **labels** for TiKV at the same time for the configurations to take effect. Otherwise, PD does not perform scheduling according to the topology.

11.15.1.3 Configure isolation-level for PD

If `location-labels` has been configured, you can further enhance the topological isolation requirements on TiKV clusters by configuring `isolation-level` in the PD configuration file.

Assume that you have made a three-layer cluster topology by configuring `location-labels` according to the instructions above: zone -> rack -> host, you can configure the `isolation-level` to `zone` as follows:

```
[replication]
isolation-level = "zone"
```

If the PD cluster is already initialized, you need to use the `pd-ctl` tool to make online changes:

```
pd-ctl config set isolation-level zone
```

The `location-level` configuration is an array of strings, which needs to correspond to a key of `location-labels`. This parameter limits the minimum and mandatory isolation level requirements on TiKV topology clusters.

Note:

`isolation-level` is empty by default, which means there is no mandatory restriction on the isolation level. To set it, you need to configure `location-labels` for PD and ensure that the value of `isolation-level` is one of `location-labels` names.

11.15.1.4 Configure a cluster using TiUP (recommended)

When using TiUP to deploy a cluster, you can configure the TiKV location in the [initialization configuration file](#). TiUP will generate the corresponding TiKV and PD configuration files during deployment.

In the following example, a two-layer topology of `zone/host` is defined. The TiKV nodes of the cluster are distributed among three zones, each zone with two hosts. In `z1`, two TiKV instances are deployed per host. In `z2` and `z3`, one TiKV instance is deployed per host. In the following example, `tikv-n` represents the IP address of the `n`th TiKV node.

```
server_configs:
  pd:
    replication.location-labels: ["zone", "host"]

tikv_servers:
## z1
```

```

- host: tikv-1
  config:
    server.labels:
      zone: z1
      host: h1
- host: tikv-2
  config:
    server.labels:
      zone: z1
      host: h1
- host: tikv-3
  config:
    server.labels:
      zone: z1
      host: h2
- host: tikv-4
  config:
    server.labels:
      zone: z1
      host: h2
## z2
- host: tikv-5
  config:
    server.labels:
      zone: z2
      host: h1
- host: tikv-6
  config:
    server.labels:
      zone: z2
      host: h2
## z3
- host: tikv-7
  config:
    server.labels:
      zone: z3
      host: h1
- host: tikv-8
  config:
    server.labels:
      zone: z3
      host: h2s

```

For details, see [Geo-distributed Deployment topology](#).

11.15.2 PD schedules based on topology label

PD schedules replicas according to the label layer to make sure that different replicas of the same data are scattered as much as possible.

Take the topology in the previous section as an example.

Assume that the number of cluster replicas is 3 (`max-replicas=3`). Because there are 3 zones in total, PD ensures that the 3 replicas of each Region are respectively placed in z1, z2, and z3. In this way, the TiDB cluster is still available when one data center fails.

Then, assume that the number of cluster replicas is 5 (`max-replicas=5`). Because there are only 3 zones in total, PD cannot guarantee the isolation of each replica at the zone level. In this situation, the PD scheduler will ensure replica isolation at the host level. In other words, multiple replicas of a Region might be distributed in the same zone but not on the same host.

In the case of the 5-replica configuration, if z3 fails or is isolated as a whole, and cannot be recovered after a period of time (controlled by `max-store-down-time`), PD will make up the 5 replicas through scheduling. At this time, only 4 hosts are available. This means that host-level isolation cannot be guaranteed and that multiple replicas might be scheduled to the same host. But if the `isolation-level` value is set to `zone` instead of being left empty, this specifies the minimum physical isolation requirements for Region replicas. That is to say, PD will ensure that replicas of the same Region are scattered among different zones. PD will not perform corresponding scheduling even if following this isolation restriction does not meet the requirement of `max-replicas` for multiple replicas.

For example, a TiKV cluster is distributed across three data zones z1, z2, and z3. Each Region has three replicas as required, and PD distributes the three replicas of the same Region to these three data zones respectively. If a power outage occurs in z1 and cannot be recovered after a period of time, PD determines that the Region replicas on z1 are no longer available. However, because `isolation-level` is set to `zone`, PD needs to strictly guarantee that different replicas of the same Region will not be scheduled on the same data zone. Because both z2 and z3 already have replicas, PD will not perform any scheduling under the minimum isolation level restriction of `isolation-level`, even if there are only two replicas at this moment.

Similarly, when `isolation-level` is set to `rack`, the minimum isolation level applies to different racks in the same data center. With this configuration, the isolation at the zone layer is guaranteed first if possible. When the isolation at the zone level cannot be guaranteed, PD tries to avoid scheduling different replicas to the same rack in the same zone. The scheduling works similarly when `isolation-level` is set to `host` where PD first guarantees the isolation level of rack, and then the level of host.

In summary, PD maximizes the disaster recovery of the cluster according to the current topology. Therefore, if you want to achieve a certain level of disaster recovery, deploy more machines on different sites according to the topology than the number of `max-replicas`. TiDB also provides mandatory configuration items such as `isolation-level` for you to more flexibly control the topological isolation level of data according to different scenarios.

12 FAQs

12.1 TiDB FAQ

This document lists the Most Frequently Asked Questions about TiDB.

12.1.1 About TiDB

12.1.1.1 TiDB introduction and architecture

12.1.1.1.1 What is TiDB?

TiDB is a distributed SQL database that features in horizontal scalability, high availability and consistent distributed transactions. It also enables you to use MySQL's SQL syntax and protocol to manage and retrieve data.

12.1.1.1.2 What is TiDB's architecture?

The TiDB cluster has three components: the TiDB server, the PD (Placement Driver) server, and the TiKV server. For more details, see [TiDB architecture](#).

12.1.1.1.3 Is TiDB based on MySQL?

No. TiDB supports MySQL syntax and protocol, but it is a new open source database that is developed and maintained by PingCAP, Inc.

12.1.1.1.4 What is the respective responsibility of TiDB, TiKV and PD (Placement Driver)?

- TiDB works as the SQL computing layer, mainly responsible for parsing SQL, specifying query plan, and generating executor.
- TiKV works as a distributed Key-Value storage engine, used to store the real data. In short, TiKV is the storage engine of TiDB.
- PD works as the cluster manager of TiDB, which manages TiKV metadata, allocates timestamps, and makes decisions for data placement and load balancing.

12.1.1.1.5 Is it easy to use TiDB?

Yes, it is. When all the required services are started, you can use TiDB as easily as a MySQL server. You can replace MySQL with TiDB to power your applications without changing a single line of code in most cases. You can also manage TiDB using the popular MySQL management tools.

12.1.1.6 How is TiDB compatible with MySQL?

Currently, TiDB supports the majority of MySQL 5.7 syntax, but does not support triggers, stored procedures, user-defined functions, and foreign keys. For more details, see [Compatibility with MySQL](#).

12.1.1.7 Does TiDB support distributed transactions?

Yes. TiDB distributes transactions across your cluster, whether it is a few nodes in a single location or many [nodes across multiple data centers](#).

Inspired by Google's Percolator, the transaction model in TiDB is mainly a two-phase commit protocol with some practical optimizations. This model relies on a timestamp allocator to assign the monotone increasing timestamp for each transaction, so conflicts can be detected. [PD](#) works as the timestamp allocator in a TiDB cluster.

12.1.1.8 What programming language can I use to work with TiDB?

Any language supported by MySQL client or driver.

12.1.1.9 Can I use other Key-Value storage engines with TiDB?

Yes. TiKV and TiDB support many popular standalone storage engines, such as GolevelDB and BoltDB. If the storage engine is a KV engine that supports transactions and it provides a client that meets the interface requirement of TiDB, then it can connect to TiDB.

12.1.1.10 In addition to the TiDB documentation, are there any other ways to acquire TiDB knowledge?

Currently [TiDB documentation](#) is the most important and timely way to get TiDB related knowledge. In addition, we also have some technical communication groups. If you have any needs, contact info@pingcap.com.

12.1.1.11 What is the length limit for the TiDB user name?

32 characters at most.

12.1.1.12 Does TiDB support XA?

No. The JDBC driver of TiDB is MySQL JDBC (Connector/J). When using Atomikos, set the data source to `type="com.mysql.jdbc.jdbc2.optional.MysqlXADataSource"`. TiDB does not support the connection with MySQL JDBC XADataSource. MySQL JDBC XADataSource only works for MySQL (for example, using DML to modify the redo log).

After you configure the two data sources of Atomikos, set the JDBC drives to XA. When Atomikos operates TM and RM (DB), Atomikos sends the command including XA to the

JDBC layer. Taking MySQL for an example, when XA is enabled in the JDBC layer, JDBC will send a series of XA logic operations to InnoDB, including using DML to change the redo log. This is the operation of the two-phase commit. The current TiDB version does not support the upper application layer JTA/XA and does not parse XA operations sent by Atomikos.

As a standalone database, MySQL can only implement across-database transactions using XA; while TiDB supports distributed transactions using Google Percolator transaction model and its performance stability is higher than XA, so TiDB does not support XA and there is no need for TiDB to support XA.

12.1.1.2 TiDB techniques

12.1.1.2.1 TiKV for data storage

See [TiDB Internal \(I\) - Data Storage](#).

12.1.1.2.2 TiDB for data computing

See [TiDB Internal \(II\) - Computing](#).

12.1.1.2.3 PD for scheduling

See [TiDB Internal \(III\) - Scheduling](#).

12.1.2 Deployment on the cloud

12.1.2.1 Public cloud

12.1.2.1.1 What cloud vendors are currently supported by TiDB?

TiDB supports deployment on [Google GKE](#), [AWS EKS](#) and [Alibaba Cloud ACK](#).

In addition, TiDB is currently available on JD Cloud and UCloud, and has the first-level database entries on them.

12.1.3 Troubleshoot

12.1.3.1 TiDB custom error messages

12.1.3.1.1 ERROR 8005 (HY000): Write Conflict, txnStartTS is stale

Check whether `tidb_disable_txn_auto_retry` is set to `on`. If so, set it to `off`; if it is already `off`, increase the value of `tidb_retry_limit` until the error no longer occurs.

12.1.3.1.2 ERROR 9001 (HY000): PD Server Timeout

A PD request timeout. Check the status, monitoring data and log of the PD server, and the network between the TiDB server and the PD server.

12.1.3.1.3 ERROR 9002 (HY000): TiKV Server Timeout

A TiKV request timeout. Check the status, monitoring data and log of the TiKV server, and the network between the TiDB server and the TiKV server.

12.1.3.1.4 ERROR 9003 (HY000): TiKV Server is Busy

The TiKV server is busy. This usually occurs when the database load is very high. Check the status, monitoring data and log of the TiKV server.

12.1.3.1.5 ERROR 9004 (HY000): Resolve Lock Timeout

A lock resolving timeout. This usually occurs when a large number of transaction conflicts exist. Check the application code to see whether lock contention exists in the database.

12.1.3.1.6 ERROR 9005 (HY000): Region is unavailable

The accessed Region is not available. A Raft Group is not available, with possible reasons like an inadequate number of replicas. This usually occurs when the TiKV server is busy or the TiKV node is shut down. Check the status, monitoring data and log of the TiKV server.

12.1.3.1.7 ERROR 9006 (HY000): GC life time is shorter than transaction duration

The interval of `GC Life Time` is too short. The data that should have been read by long transactions might be deleted. You can adjust `tidb_gc_life_time` using the following command:

```
SET GLOBAL tidb_gc_life_time = '30m';
```

Note:

“30m” means only cleaning up the data generated 30 minutes ago, which might consume some extra storage space.

12.1.3.1.8 ERROR 9007 (HY000): Write Conflict

Check whether `tidb_disable_txn_auto_retry` is set to `on`. If so, set it to `off`; if it is already `off`, increase the value of `tidb_retry_limit` until the error no longer occurs.

12.1.3.1.9 ERROR 8130 (HY000): client has multi-statement capability disabled

This error might occur after upgrading from an earlier version of TiDB. To reduce the impact of SQL injection attacks, TiDB now prevents multiple queries from being executed in the same `COM_QUERY` call by default.

The system variable `tidb_multi_statement_mode` can be used to control this behavior.

12.1.3.2 MySQL native error messages

12.1.3.2.1 ERROR 2013 (HY000): Lost connection to MySQL server during query

- Check whether panic is in the log.
- Check whether OOM exists in dmesg using `dmesg -T | grep -i oom`.
- A long time of no access might also lead to this error. It is usually caused by TCP timeout. If TCP is not used for a long time, the operating system kills it.

12.1.3.2.2 ERROR 1105 (HY000): other error: unknown error Wire Error(InvalidEnumValue(4004))

This error usually occurs when the version of TiDB does not match with the version of TiKV. To avoid version mismatch, upgrade all components when you upgrade the version.

12.1.3.2.3 ERROR 1148 (42000): the used command is not allowed with this TiDB version

When you execute the `LOAD DATA LOCAL` statement but the MySQL client does not allow executing this statement (the value of the `local_infile` option is 0), this error occurs.

The solution is to use the `--local-infile=1` option when you start the MySQL client. For example, use command like `mysql --local-infile=1 -u root -h 127.0.0.1 -P 4000`. The default value of `local-infile` is different in different versions of MySQL client, therefore you need to configure it in some MySQL clients and do not need to configure it in some others.

12.1.3.2.4 ERROR 9001 (HY000): PD server timeout start timestamp may fall behind safe point

This error occurs when TiDB fails to access PD. A worker in the TiDB background continuously queries the safepoint from PD and this error occurs if it fails to query within 100s. Generally, it is because the disk on PD is slow and busy or the network failed between TiDB and PD. For the details of common errors, see [Error Number and Fault Diagnosis](#).

12.1.3.3 TiDB log error messages

12.1.3.3.1 EOF error

When the client or proxy disconnects from TiDB, TiDB does not immediately notice that the connection has been disconnected. Instead, TiDB can only notice the disconnection when it begins to return data to the connection. At this time, the log prints an EOF error.

12.2 SQL FAQs

This document summarizes the FAQs related to SQL operations in TiDB.

12.2.1 What are the MySQL variables that TiDB is compatible with?

See [System Variables](#).

12.2.2 The order of results is different from MySQL when ORDER BY is omitted

It is not a bug. The default order of records depends on various situations without any guarantee of consistency.

The order of results in MySQL might appear stable because queries are executed in a single thread. However, it is common that query plans can change when upgrading to new versions. It is recommended to use `ORDER BY` whenever an order of results is desired.

The reference can be found in [ISO/IEC 9075:1992, Database Language SQL- July 30, 1992](#), which states as follows:

If an `<order by clause>` is not specified, then the table specified by the `<cursor specification>` is T and the ordering of rows in T is implementation-dependent.

In the following two queries, both results are considered legal:

```
> select * from t;
+---+---+
| a | b |
+---+---+
| 1 | 1 |
| 2 | 2 |
+---+---+
2 rows in set (0.00 sec)
```

```
> select * from t; -- the order of results is not guaranteed
+-----+
| a    | b    |
+-----+
| 2    | 2    |
| 1    | 1    |
+-----+
2 rows in set (0.00 sec)
```

A statement is also considered non-deterministic if the list of columns used in the ORDER BY is non-unique. In the following example, the column a has duplicate values. Thus, only ORDER BY a, b would be guaranteed deterministic:

```
> select * from t order by a;
+-----+
| a    | b    |
+-----+
| 1    | 1    |
| 2    | 1    |
| 2    | 2    |
+-----+
3 rows in set (0.00 sec)
```

```
> select * from t order by a; -- the order of column a is guaranteed, but
   ↪ b is not
+-----+
| a    | b    |
+-----+
| 1    | 1    |
| 2    | 2    |
| 2    | 1    |
+-----+
3 rows in set (0.00 sec)
```

12.2.3 Does TiDB support SELECT FOR UPDATE?

Yes. When using pessimistic locking (the default since TiDB v3.0) the SELECT FOR UPDATE execution behaves similar to MySQL.

When using optimistic locking, SELECT FOR UPDATE does not lock data when the transaction is started, but checks conflicts when the transaction is committed. If the check reveals conflicts, the committing transaction rolls back.

12.2.4 Can the codec of TiDB guarantee that the UTF-8 string is memcomparable? Is there any coding suggestion if our key needs to support UTF-8?

TiDB uses the UTF-8 character set by default and currently only supports UTF-8. The string of TiDB uses the memcomparable format.

12.2.5 What is the maximum number of statements in a transaction?

The maximum number of statements in a transaction is 5000 by default.

12.2.6 Why does the auto-increment ID of the later inserted data is smaller than that of the earlier inserted data in TiDB?

The auto-increment ID feature in TiDB is only guaranteed to be automatically incremental and unique but is not guaranteed to be allocated sequentially. Currently, TiDB is allocating IDs in batches. If data is inserted into multiple TiDB servers simultaneously, the allocated IDs are not sequential. When multiple threads concurrently insert data to multiple `tidb-server` instances, the auto-increment ID of the later inserted data may be smaller. TiDB allows specifying `AUTO_INCREMENT` for the integer field, but allows only one `AUTO_INCREMENT` field in a single table. For details, see [Auto-increment ID](#).

12.2.7 How do I modify the `sql_mode` in TiDB?

TiDB supports modifying the `sql_mode` system variables on a SESSION or GLOBAL basis. Changes to `GLOBAL` scoped variables propagate to the rest servers of the cluster and persist across restarts. This means that you do not need to change the `sql_mode` value on each TiDB server.

12.2.8 Error: `java.sql.BatchUpdateException:statement count 5001 exceeds the transaction limitation while using Sqoop to write data into TiDB in batches`

In Sqoop, `--batch` means committing 100 statements in each batch, but by default each statement contains 100 SQL statements. So, $100 * 100 = 10000$ SQL statements, which exceeds 5000, the maximum number of statements allowed in a single TiDB transaction.

Two solutions:

- Add the `-Dsqoop.export.records.per.statement=10` option as follows:

```
sqoop export \
  -Dsqoop.export.records.per.statement=10 \
  --connect jdbc:mysql://mysql.example.com/sqoop \
  --username sqoop ${user} \
```

```
--password ${passwd} \
--table ${tab_name} \
--export-dir ${dir} \
--batch
```

- You can also increase the limited number of statements in a single TiDB transaction, but this will consume more memory.

12.2.9 Does TiDB have a function like the Flashback Query in Oracle? Does it support DDL?

Yes, it does. And it supports DDL as well. For details, see [how TiDB reads data from history versions](#).

12.2.10 Does TiDB release space immediately after deleting data?

None of the `DELETE`, `TRUNCATE` and `DROP` operations release data immediately. For the `TRUNCATE` and `DROP` operations, after the TiDB GC (Garbage Collection) time (10 minutes by default), the data is deleted and the space is released. For the `DELETE` operation, the data is deleted but the space is not released according to TiDB GC. When subsequent data is written into RocksDB and executes `COMPACT`, the space is reused.

12.2.11 Does TiDB support the `REPLACE INTO` syntax?

Yes. The exception being that `LOAD DATA` does not currently support the `REPLACE INTO` syntax.

12.2.12 Why does the query speed get slow after data is deleted?

Deleting a large amount of data leaves a lot of useless keys, affecting the query efficiency. Currently the [Region Merge](#) feature is in development, which is expected to solve this problem. For details, see the [deleting data section in TiDB Best Practices](#).

12.2.13 What should I do if it is slow to reclaim storage space after deleting data?

Because TiDB uses Multiversion concurrency control (MVCC), deleting data does not immediately reclaim space. Garbage collection is delayed so that concurrent transactions are able to see earlier versions of rows. This can be configured via the `tidb_gc_life_time` (default: 10m0s) system variable.

12.2.14 Does SHOW PROCESSLIST display the system process ID?

The display content of TiDB SHOW PROCESSLIST is almost the same as that of MySQL SHOW PROCESSLIST. TiDB show processlist does not display the system process ID. The ID that it displays is the current session ID. The differences between TiDB show → processlist and MySQL show processlist are as follows:

- As TiDB is a distributed database, the `tidb-server` instance is a stateless engine for parsing and executing the SQL statements (for details, see [TiDB architecture](#)). `show → processlist` displays the session list executed in the `tidb-server` instance that the user logs in to from the MySQL client, not the list of all the sessions running in the cluster. But MySQL is a standalone database and its `show processlist` displays all the SQL statements executed in MySQL.
- The `State` column in TiDB is not continually updated during query execution. As TiDB supports parallel query, each statement may be in multiple *states* at once, and thus it is difficult to simplify to a single value.

12.2.15 How to control or change the execution priority of SQL commits?

TiDB supports changing the priority on a `per-session`, `global` or individual statement basis. Priority has the following meaning:

- `HIGH_PRIORITY`: this statement has a high priority, that is, TiDB gives priority to this statement and executes it first.
- `LOW_PRIORITY`: this statement has a low priority, that is, TiDB reduces the priority of this statement during the execution period.

You can combine the above two parameters with the DML of TiDB to use them. For example:

1. Adjust the priority by writing SQL statements in the database:

```
select HIGH_PRIORITY | LOW_PRIORITY count(*) from table_name;
insert HIGH_PRIORITY | LOW_PRIORITY into table_name insert_values;
delete HIGH_PRIORITY | LOW_PRIORITY from table_name;
update HIGH_PRIORITY | LOW_PRIORITY table_reference set assignment_list
    → where where_condition;
replace HIGH_PRIORITY | LOW_PRIORITY into table_name;
```

2. The full table scan statement automatically adjusts itself to a low priority. `analyze` has a low priority by default.

12.2.16 What's the trigger strategy for auto analyze in TiDB?

Trigger strategy: `auto analyze` is automatically triggered when the number of pieces of data in a new table reaches 1000 and this table has no write operation within one minute.

When the modified number or the current total row number is larger than `tidb_auto_analyze_ratio`, the `analyze` statement is automatically triggered. The default value of `tidb_auto_analyze_ratio` is 0.5, indicating that this feature is enabled by default. To ensure safety, its minimum value is 0.3 when the feature is enabled, and it must be smaller than `pseudo-estimate-ratio` whose default value is 0.8, otherwise pseudo statistics will be used for a period of time. It is recommended to set `tidb_auto_analyze_ratio` to 0.5.

12.2.17 Can I use hints to override the optimizer behavior?

TiDB supports multiple ways to override the default query optimizer behavior, including `hints` and [SQL Plan Management](#). The basic usage is similar to MySQL, with several TiDB specific extensions:

```
SELECT column_name FROM table_name USE INDEX ( index_name ) WHERE
    ↪ where_condition;
```

12.2.18 Why the Information schema is changed error is reported?

TiDB handles the SQL statement using the `schema` of the time and supports online asynchronous DDL change. A DML statement and a DDL statement might be executed at the same time and you must ensure that each statement is executed using the same `schema`. Therefore, when the DML operation meets the ongoing DDL operation, the `Information schema is changed` error might be reported. Some improvements have been made to prevent too many error reportings during the DML operation.

Now, there are still a few reasons for this error reporting (only the first one is related to tables):

- Some tables involved in the DML operation are the same tables involved in the ongoing DDL operation.
- The DML operation goes on for a long time. During this period, many DDL statements have been executed, which causes more than 1024 `schema` version changes. You can modify this default value by modifying the `tidb_max_delta_schema_count` variable.
- The TiDB server that accepts the DML request is not able to load `schema information` for a long time (possibly caused by the connection failure between TiDB and PD or TiKV). During this period, many DDL statements have been executed, which causes more than 100 `schema` version changes.
- After TiDB restarts and before the first DDL operation is executed, the DML operation is executed and then encounters the first DDL operation (which means before the first

DDL operation is executed, the transaction corresponding to the DML is started. And after the first `schema` version of the DDL is changed, the transaction corresponding to the DML is committed), this DML operation reports this error.

Note:

- Currently, TiDB does not cache all the `schema` version changes.
- For each DDL operation, the number of `schema` version changes is the same with the number of corresponding `schema state` version changes.
- Different DDL operations cause different number of `schema` version changes. For example, the `CREATE TABLE` statement causes one `schema` version change while the `ADD COLUMN` statement causes four.

12.2.19 What are the causes of the “Information schema is out of date” error?

When executing a DML statement, if TiDB fails to load the latest schema within a DDL lease (45s by default), the `Information schema is out of date` error might occur. Possible causes are:

- The TiDB instance that executed this DML was killed, and the transaction execution corresponding to this DML statement took longer than a DDL lease. When the transaction was committed, the error occurred.
- TiDB failed to connect to PD or TiKV while executing this DML statement. As a result, TiDB failed to load schema within a DDL lease or disconnected from PD due to the keepalive setting.

12.2.20 Error is reported when executing DDL statements under high concurrency?

When you execute DDL statements (such as creating tables in batches) under high concurrency, a very few of these statements might fail because of key conflicts during the concurrent execution.

It is recommended to keep the number of concurrent DDL statements under 20. Otherwise, you need to retry the failed statements from the client.

12.2.21 SQL optimization

12.2.21.1 TiDB execution plan description

See [Understand the Query Execution Plan](#).

12.2.21.2 Statistics collection

See [Introduction to Statistics](#).

12.2.21.3 How to optimize `select count(1)`?

The `count(1)` statement counts the total number of rows in a table. Improving the degree of concurrency can significantly improve the speed. To modify the concurrency, refer to the [document](#). But it also depends on the CPU and I/O resources. TiDB accesses TiKV in every query. When the amount of data is small, all MySQL is in memory, and TiDB needs to conduct a network access.

Recommendations:

1. Improve the hardware configuration. See [Software and Hardware Requirements](#).
2. Improve the concurrency. The default value is 10. You can improve it to 50 and have a try. But usually the improvement is 2-4 times of the default value.
3. Test the `count` in the case of large amount of data.
4. Optimize the TiKV configuration. See [Tune TiKV Thread Performance](#) and [Tune TiKV Memory Performance](#).
5. Enable the [Coprocessor Cache](#).

12.2.21.4 How to view the progress of the current DDL job?

You can use `admin show ddl` to view the progress of the current DDL job. The operation is as follows:

```
admin show ddl;
```

```
*****
1. row *****
SCHEMA_VER: 140
OWNER: 1a1c4174-0fcf-4ba0-add9-12d08c4077dc
RUNNING_JOBS: ID:121, Type:add index, State:running, SchemaState:write
    ↳ reorganization, SchemaID:1, TableID:118, RowCount:77312, ArgLen:0,
    ↳ start time: 2018-12-05 16:26:10.652 +0800 CST, Err:<nil>, ErrCount:0,
    ↳ SnapshotVersion:404749908941733890
SELF_ID: 1a1c4174-0fcf-4ba0-add9-12d08c4077dc
```

From the above results, you can get that the `add index` operation is being processed currently. You can also get from the `RowCount` field of the `RUNNING_JOBS` column that now the `add index` operation has added 77312 rows of indexes.

12.2.21.5 How to view the DDL job?

- `admin show ddl`: to view the running DDL job

- `admin show ddl jobs`: to view all the results in the current DDL job queue (including tasks that are running and waiting to run) and the last ten results in the completed DDL job queue
- `admin show ddl job queries 'job_id' [, 'job_id'] ...`: to view the original SQL statement of the DDL task corresponding to the `job_id`; the `job_id` only searches the running DDL job and the last ten results in the DDL history job queue.

12.2.21.6 Does TiDB support CBO (Cost-Based Optimization)? If yes, to what extent?

Yes. TiDB uses the cost-based optimizer. The cost model and statistics are constantly optimized. TiDB also supports join algorithms like hash join and sort-merge join.

12.2.21.7 How to determine whether I need to execute `analyze` on a table?

View the `Healthy` field using `show stats_healthy` and generally you need to execute `analyze` on a table when the field value is smaller than 60.

12.2.21.8 What is the ID rule when a query plan is presented as a tree? What is the execution order for this tree?

No rule exists for these IDs but the IDs are unique. When IDs are generated, a counter works and adds one when one plan is generated. The execution order has nothing to do with the ID. The whole query plan is a tree and the execution process starts from the root node and the data is returned to the upper level continuously. For details about the query plan, see [Understanding the TiDB Query Execution Plan](#).

12.2.21.9 In the TiDB query plan, `cop` tasks are in the same root. Are they executed concurrently?

Currently the computing tasks of TiDB belong to two different types of tasks: `cop task` and `root task`.

`cop task` is the computing task which is pushed down to the KV end for distributed execution; `root task` is the computing task for single point execution on the TiDB end.

Generally the input data of `root task` comes from `cop task`; when `root task` processes data, `cop task` of TiKV can processes data at the same time and waits for the pull of `root task` of TiDB. Therefore, `cop` tasks can be considered as executed concurrently; but their data has an upstream and downstream relationship. During the execution process, they are executed concurrently during some time. For example, the first `cop task` is processing the data in [100, 200] and the second `cop task` is processing the data in [1, 100]. For details, see [Understanding the TiDB Query Plan](#).

12.2.22 Database optimization

12.2.22.1 Edit TiDB options

See [The TiDB Command Options](#).

12.2.22.2 How to scatter the hotspots?

In TiDB, data is divided into Regions for management. Generally, the TiDB hotspot means the Read/Write hotspot in a Region. In TiDB, for the table whose primary key (PK) is not an integer or which has no PK, you can properly break Regions by configuring SHARD_ROW_ID_BITS to scatter the Region hotspots. For details, see the introduction of SHARD_ROW_ID_BITS in [SHARD_ROW_ID_BITS](#).

12.2.22.3 Tune TiKV performance

See [Tune TiKV Thread Performance](#) and [Tune TiKV Memory Performance](#).

12.3 Deployment, Operations and Maintenance FAQs

This document summarizes the FAQs related to TiDB deployment, operations and maintenance.

12.3.1 Operating system requirements

12.3.1.1 What are the required operating system versions?

Linux OS Platform	Version
Red Hat Enterprise Linux	7.3 or later
CentOS	7.3 or later
Oracle Enterprise Linux	7.3 or later

12.3.1.2 Why it is recommended to deploy the TiDB cluster on CentOS 7?

As an open source distributed NewSQL database with high performance, TiDB can be deployed in the Intel architecture server and major virtualization environments and runs well. TiDB supports most of the major hardware networks and Linux operating systems. For details, see [Official Deployment Requirements](#) for deploying TiDB.

A lot of TiDB tests have been carried out in CentOS 7.3, and many deployment best practices have been accumulated in CentOS 7.3. Therefore, it is recommended that you use the CentOS 7.3+ Linux operating system when deploying TiDB.

12.3.2 Server requirements

You can deploy and run TiDB on the 64-bit generic hardware server platform in the Intel x86-64 architecture. The requirements and recommendations about server hardware configuration for development, testing and production environments are as follows:

12.3.2.1 Development and testing environments

Component	CPU	Memory	Local Storage	Network	Instance Number (Minimum Requirement)
TiDB	8 core+	16 GB+	SAS, 200 GB+	Gigabit network card	1 (can be deployed on the same machine with PD)
PD	8 core+	16 GB+	SAS, 200 GB+	Gigabit network card	1 (can be deployed on the same machine with TiDB)
TiKV	8 core+	32 GB+	SAS, 200 GB+	Gigabit network card	3
				Total Server Number	4

12.3.2.2 Production environment

Component	CPU	Memory	Hard Disk Type	Network	Instance N
TiDB	16 core+	48 GB+	SAS	10 Gigabit network card (2 preferred)	
PD	8 core+	16 GB+	SSD	10 Gigabit network card (2 preferred)	
TiKV	16 core+	48 GB+	SSD	10 Gigabit network card (2 preferred)	
Monitor	8 core+	16 GB+	SAS	Gigabit network card Total Server Number	

12.3.2.3 What's the purposes of 2 network cards of 10 gigabit?

As a distributed cluster, TiDB has a high demand on time, especially for PD, because PD needs to distribute unique timestamps. If the time in the PD servers is not consistent, it takes longer waiting time when switching the PD server. The bond of two network cards guarantees the stability of data transmission, and 10 gigabit guarantees the transmission speed. Gigabit network cards are prone to meet bottlenecks, therefore it is strongly recommended to use 10 gigabit network cards.

12.3.2.4 Is it feasible if we don't use RAID for SSD?

If the resources are adequate, it is recommended to use RAID 10 for SSD. If the resources are inadequate, it is acceptable not to use RAID for SSD.

12.3.2.5 What's the recommended configuration of TiDB components?

- TiDB has a high requirement on CPU and memory. If you need to enable TiDB Binlog, the local disk space should be increased based on the service volume estimation and the time requirement for the GC operation. But the SSD disk is not a must.
- PD stores the cluster metadata and has frequent Read and Write requests. It demands a high I/O disk. A disk of low performance will affect the performance of the whole cluster. It is recommended to use SSD disks. In addition, a larger number of Regions has a higher requirement on CPU and memory.
- TiKV has a high requirement on CPU, memory and disk. It is required to use SSD.

For details, see [Software and Hardware Recommendations](#).

12.3.3 Installation and deployment

For the production environment, it is recommended to use [TiUP](#) to deploy your TiDB cluster. See [Deploy a TiDB Cluster Using TiUP](#).

12.3.3.1 Why the modified `toml` configuration for TiKV/PD does not take effect?

You need to set the `--config` parameter in TiKV/PD to make the `toml` configuration effective. TiKV/PD does not read the configuration by default. Currently, this issue only occurs when deploying using Binary. For TiKV, edit the configuration and restart the service. For PD, the configuration file is only read when PD is started for the first time, after which you can modify the configuration using `pd-ctl`. For details, see [PD Control User Guide](#).

12.3.3.2 Should I deploy the TiDB monitoring framework (Prometheus + Grafana) on a standalone machine or on multiple machines? What is the recommended CPU and memory?

The monitoring machine is recommended to use standalone deployment. It is recommended to use an 8 core CPU with 16 GB+ memory and a 500 GB+ hard disk.

12.3.3.3 Why the monitor cannot display all metrics?

Check the time difference between the machine time of the monitor and the time within the cluster. If it is large, you can correct the time and the monitor will display all the metrics.

12.3.3.4 What is the function of `supervise`/`svc`/`svstat` service?

- `supervise`: the daemon process, to manage the processes
- `svc`: to start and stop the service
- `svstat`: to check the process status

12.3.3.5 Description of inventory.ini variables

Variable	Description
<code>cluster_name</code>	name of a cluster, adjustable
<code>tidb_version</code>	version of TiDB
<code>deployment_method</code>	method of deployment, binary by default, Docker optional
<code>processes_supervision</code>	way of processes, systemd by default, supervise optional

Variable	Description
<code>timezone</code>	
↳	time- zone of the man- aged node, ad- justable, <code>Asia/</code> ↳ <code>Shanghai</code> ↳ by default, used with the <code>set_timezone</code> ↳ variable
<code>set_timezone</code>	
↳	the time- zone of the man- aged node, True by default; False means closing
<code>enable_ceilently</code>	
↳	not sup- ported
<code>enable_tfirewalld</code>	
↳	enable the firewall, closed by default

Variable	Description
----------	-------------

<code>enable_ntpd</code>	
↪	monitor the NTP service of the man- aged node, True by default; do not close it
<code>machine_to_benchmark</code>	
↪	monitor the disk IOPS of the man- aged node, True by default; do not close it
<code>set_hostname</code>	
↪	the host- name of the man- aged node based on the IP, False by default

Variable	Description
<code>enable_binlog</code>	→ to deploy Pump and enable the binlog, False by default, dependent on the Kafka cluster; see the <code>zookeeper_addrs</code>
<code>zookeeper_addr</code>	→ variable <code>zookeeper_addr</code> → ZooKeeper address of the binlog Kafka cluster

VariableDescription

`enable_slow_query_log`
↪ record
the slow
query
log of
TiDB
into a
single
file: ({{
de-
ploy_dir
}}/log/tidb_slow_query.log).
False by
default,
to
record
it into
the
TiDB
log

Variable	Description
deploy_without_tidb	
↳	Key-Value mode, deploy only PD, TiKV and the monitoring service, not TiDB; set the IP of the tidb_servers host group to null in the inventory
↳ .	
↳ ini	
file	

12.3.3.6 How to separately record the slow query log in TiDB? How to locate the slow query SQL statement?

1. The slow query definition for TiDB is in the TiDB configuration file. The `slow-threshold: 300` parameter is used to configure the threshold value of the slow query (unit: millisecond).
2. If a slow query occurs, you can locate the `tidb-server` instance where the slow query is and the slow query time point using Grafana and find the SQL statement information recorded in the log on the corresponding node.
3. In addition to the log, you can also view the slow query using the `admin show slow` command. For details, see [admin show slow command](#).

12.3.3.7 How to add the label configuration if label of TiKV was not configured when I deployed the TiDB cluster for the first time?

The configuration of TiDB `label` is related to the cluster deployment architecture. It is important and is the basis for PD to execute global management and scheduling. If you did not configure `label` when deploying the cluster previously, you should adjust the deployment structure by manually adding the `location-labels` information using the PD management tool `pd-ctl`, for example, `config set location-labels "zone,rack,host"` (you should configure it based on the practical `label` level name).

For the usage of `pd-ctl`, see [PD Control User Guide](#).

12.3.3.8 Why does the `dd` command for the disk test use the `oflag=direct` option?

The Direct mode wraps the Write request into the I/O command and sends this command to the disk to bypass the file system cache and directly test the real I/O Read/Write performance of the disk.

12.3.3.9 How to use the `fio` command to test the disk performance of the TiKV instance?

- Random Read test:

```
./fio -ioengine=psync -bs=32k -fdatasync=1 -thread -rw=randread -size
    ↵ =10G -filename=fio_randread_test.txt -name='fio randread test' -
    ↵ iodepth=4 -runtime=60 -numjobs=4 -group_reporting --output-format
    ↵ =json --output=fio_randread_result.json
```

- The mix test of sequential Write and random Read:

```
./fio -ioengine=psync -bs=32k -fdatasync=1 -thread -rw=randrw -
    ↵ percentage_random=100,0 -size=10G -filename=
    ↵ fio_randread_write_test.txt -name='fio mixed randread and
    ↵ sequential write test' -iodepth=4 -runtime=60 -numjobs=4 -
    ↵ group_reporting --output-format=json --output=
    ↵ fio_randread_write_test.json
```

12.3.4 Cluster management

12.3.4.1 Daily management

12.3.4.1.1 How to log into TiDB?

You can log into TiDB like logging into MySQL. For example:

```
mysql -h 127.0.0.1 -uroot -P4000
```

12.3.4.1.2 How to modify the system variables in TiDB?

Similar to MySQL, TiDB includes static and solid parameters. You can directly modify static parameters using `set global xxx = n`, but the new value of a parameter is only effective within the life cycle in this instance.

12.3.4.1.3 Where and what are the data directories in TiDB (TiKV)?

TiKV data is located in the `--data-dir`, which include four directories of backup, db, raft, and snap, used to store backup, data, Raft data, and mirror data respectively.

12.3.4.1.4 What are the system tables in TiDB?

Similar to MySQL, TiDB includes system tables as well, used to store the information required by the server when it runs. See [TiDB system table](#).

12.3.4.1.5 Where are the TiDB/PD/TiKV logs?

By default, TiDB/PD/TiKV outputs standard error in the logs. If a log file is specified by `--log-file` during the startup, the log is output to the specified file and executes rotation daily.

12.3.4.1.6 How to safely stop TiDB?

Kill all the services using `kill` directly. The components of TiDB will do `graceful shutdown`.

12.3.4.1.7 Can `kill` be executed in TiDB?

- You can `kill` DML statements. First use `show processlist` to find the ID corresponding with the session, and then run `kill tidb [session id]`.
- You can `kill` DDL statements. First use `admin show ddl jobs` to find the ID of the DDL job you need to kill, and then run `admin cancel ddl jobs 'job_id' [, → 'job_id']` For more details, see the [ADMIN statement](#).

12.3.4.1.8 Does TiDB support session timeout?

TiDB does not currently support session timeout at the database level. At present, if you want to achieve timeout, when there is no LB (Load Balancing), you need to record the ID of the initiated Session on the application side. You can customize the timeout through the application. After timeout, you need to go to the node that initiated the Query Use `kill tidb [session id]` to kill SQL. It is currently recommended to use an application program to achieve session timeout. When the timeout period is reached, the application layer will throw an exception and continue to execute subsequent program segments.

12.3.4.1.9 What is the TiDB version management strategy for production environment? How to avoid frequent upgrade?

Currently, TiDB has a standard management of various versions. Each release contains a detailed change log and [release notes](#). Whether it is necessary to upgrade in the production environment depends on the application system. It is recommended to learn the details about the functional differences between the previous and later versions before upgrading.

Take Release Version: v1.0.3-1-ga80e796 as an example of version number description:

- v1.0.3 indicates the standard GA version.
- -1 indicates the current version has one commit.
- ga80e796 indicates the version git-hash.

12.3.4.1.10 What's the difference between various TiDB master versions?

The TiDB community is highly active. After the 1.0 GA release, the engineers have been keeping optimizing and fixing bugs. Therefore, the TiDB version is updated quite fast. If you want to keep informed of the latest version, see [TiDB Weekly update](#).

It is recommended to [deploy TiDB using TiUP](#). TiDB has a unified management of the version number after the 1.0 GA release. You can view the version number using the following two methods:

- `select tidb_version()`
- `tidb-server -V`

12.3.4.1.11 Is there a graphical deployment tool for TiDB?

Currently no.

12.3.4.1.12 How to scale TiDB horizontally?

As your business grows, your database might face the following three bottlenecks:

- Lack of storage resources which means that the disk space is not enough.
- Lack of computing resources such as high CPU occupancy.
- Not enough write and read capacity.

You can scale TiDB as your business grows.

- If the disk space is not enough, you can increase the capacity simply by adding more TiKV nodes. When the new node is started, PD will migrate the data from other nodes to the new node automatically.

- If the computing resources are not enough, check the CPU consumption situation first before adding more TiDB nodes or TiKV nodes. When a TiDB node is added, you can configure it in the Load Balancer.
- If the capacity is not enough, you can add both TiDB nodes and TiKV nodes.

12.3.4.1.13 If Percolator uses distributed locks and the crash client keeps the lock, will the lock not be released?

For more details, see [Percolator and TiDB Transaction Algorithm](#) in Chinese.

12.3.4.1.14 Why does TiDB use gRPC instead of Thrift? Is it because Google uses it?

Not really. We need some good features of gRPC, such as flow control, encryption and streaming.

12.3.4.1.15 What does the 92 indicate in `like(bind0.customers.name, jason%, 92)`?

The 92 indicates the escape character, which is ASCII 92 by default.

12.3.4.1.16 Why does the data length shown by `information_schema.tables.data_length` differ from the store size on the TiKV monitoring panel?

Two reasons:

- The two results are calculated in different ways. `information_schema.tables → data_length` is an estimated value by calculating the averaged length of each row, while the store size on the TiKV monitoring panel sums up the length of the data files (the SST files of RocksDB) in a single TiKV instance.
- `information_schema.tables.data_length` is a logical value, while the store size is a physical value. The redundant data generated by multiple versions of the transaction is not included in the logical value, while the redundant data is compressed by TiKV in the physical value.

12.3.4.1.17 Why does the transaction not use the Async Commit or the one-phase commit feature?

In the following situations, even you have enabled the **Async Commit** feature and the **one-phase commit** feature using the system variables, TiDB will not use these features:

- If you have enabled TiDB Binlog, restricted by the implementation of TiDB Binlog, TiDB does not use the Async Commit or one-phase commit feature.
- TiDB uses the Async Commit or one-phase commit features only when no more than 256 key-value pairs are written in the transaction and the total size of keys is no more than 4 KB. This is because, for transactions with a large amount of data to write, using Async Commit cannot greatly improve the performance.

12.3.4.2 PD management

12.3.4.2.1 The TiKV cluster is not bootstrapped message is displayed when I access PD

Most of the APIs of PD are available only when the TiKV cluster is initialized. This message is displayed if PD is accessed when PD is started while TiKV is not started when a new cluster is deployed. If this message is displayed, start the TiKV cluster. When TiKV is initialized, PD is accessible.

12.3.4.2.2 The etcd cluster ID mismatch message is displayed when starting PD

This is because the `--initial-cluster` in the PD startup parameter contains a member that doesn't belong to this cluster. To solve this problem, check the corresponding cluster of each member, remove the wrong member, and then restart PD.

12.3.4.2.3 What's the maximum tolerance for time synchronization error of PD?

PD can tolerate any synchronization error, but a larger error value means a larger gap between the timestamp allocated by the PD and the physical time, which will affect functions such as read of historical versions.

12.3.4.2.4 How does the client connection find PD?

The client connection can only access the cluster through TiDB. TiDB connects PD and TiKV. PD and TiKV are transparent to the client. When TiDB connects to any PD, the PD tells TiDB who is the current leader. If this PD is not the leader, TiDB reconnects to the leader PD.

12.3.4.2.5 What is the difference between the `leader-schedule-limit` and `region-schedule-limit` scheduling parameters in PD?

- The `leader-schedule-limit` scheduling parameter is used to balance the Leader number of different TiKV servers, affecting the load of query processing.
- The `region-schedule-limit` scheduling parameter is used to balance the replica number of different TiKV servers, affecting the data amount of different nodes.

12.3.4.2.6 Is the number of replicas in each region configurable? If yes, how to configure it?

Yes. Currently, you can only update the global number of replicas. When started for the first time, PD reads the configuration file (`conf/pd.yml`) and uses the `max-replicas` configuration in it. If you want to update the number later, use the `pd-ctl` configuration command

`config set max-replicas $num` and view the enabled configuration using `config show → all`. The updating does not affect the applications and is configured in the background.

Make sure that the total number of TiKV instances is always greater than or equal to the number of replicas you set. For example, 3 replicas need 3 TiKV instances at least. Additional storage requirements need to be estimated before increasing the number of replicas. For more information about pd-ctl, see [PD Control User Guide](#).

12.3.4.2.7 How to check the health status of the whole cluster when lacking command line cluster management tools?

You can determine the general status of the cluster using the pd-ctl tool. For detailed cluster status, you need to use the monitor to determine.

12.3.4.2.8 How to delete the monitoring data of a cluster node that is offline?

The offline node usually indicates the TiKV node. You can determine whether the offline process is finished by the pd-ctl or the monitor. After the node is offline, perform the following steps:

1. Manually stop the relevant services on the offline node.
2. Delete the `node_exporter` data of the corresponding node from the Prometheus configuration file.

12.3.4.3 TiDB server management

12.3.4.3.1 How to set the lease parameter in TiDB?

The lease parameter (`--lease=60`) is set from the command line when starting a TiDB server. The value of the lease parameter impacts the Database Schema Changes (DDL) speed of the current session. In the testing environments, you can set the value to 1s for to speed up the testing cycle. But in the production environments, it is recommended to set the value to minutes (for example, 60) to ensure the DDL safety.

12.3.4.3.2 What is the processing time of a DDL operation?

The processing time is different for different scenarios. Generally, you can consider the following three scenarios:

1. The `Add Index` operation with a relatively small number of rows in the corresponding data table: about 3s
2. The `Add Index` operation with a relatively large number of rows in the corresponding data table: the processing time depends on the specific number of rows and the QPS at that time (the `Add Index` operation has a lower priority than ordinary SQL operations)

3. Other DDL operations: about 1s

If the TiDB server instance that receives the DDL request is the same TiDB server instance that the DDL owner is in, the first and third scenarios above may cost only dozens to hundreds of milliseconds.

12.3.4.3.3 Why it is very slow to run DDL statements sometimes?

Possible reasons:

- If you run multiple DDL statements together, the last few DDL statements might run slowly. This is because the DDL statements are executed serially in the TiDB cluster.
- After you start the cluster successfully, the first DDL operation may take a longer time to run, usually around 30s. This is because the TiDB cluster is electing the leader that processes DDL statements.
- The processing time of DDL statements in the first ten minutes after starting TiDB would be much longer than the normal case if you meet the following conditions: 1) TiDB cannot communicate with PD as usual when you are stopping TiDB (including the case of power failure); 2) TiDB fails to clean up the registration data from PD in time because TiDB is stopped by the `kill -9` command. If you run DDL statements during this period, for the state change of each DDL, you need to wait for $2 * \text{lease}$ ($\text{lease} = 45\text{s}$).
- If a communication issue occurs between a TiDB server and a PD server in the cluster, the TiDB server cannot get or update the version information from the PD server in time. In this case, you need to wait for $2 * \text{lease}$ for the state processing of each DDL.

12.3.4.3.4 Can I use S3 as the backend storage engine in TiDB?

No. Currently, TiDB only supports the distributed storage engine and the Goleveldb/RocksDB/BoltDB engine.

12.3.4.3.5 Can the `Information_schema` support more real information?

As part of MySQL compatibility, TiDB supports a number of `INFORMATION_SCHEMA` → tables. Many of these tables also have a corresponding SHOW command. For more information, see [Information Schema](#).

12.3.4.3.6 What's the explanation of the TiDB Backoff type scenario?

In the communication process between the TiDB server and the TiKV server, the Server → is busy or `backoff.maxsleep 20000ms` log message is displayed when processing a large volume of data. This is because the system is busy while the TiKV server processes data. At this time, usually you can view that the TiKV host resources usage rate is high. If this occurs, you can increase the server capacity according to the resources usage.

12.3.4.3.7 What is the main reason of TiDB TiClient type?

The TiClient Region Error indicator describes the error types and metrics that appear when the TiDB server acts as a client to access the TiKV server through the KV interface to perform data operations. The error types include `not_leader` and `stale_epoch`. These errors occur when the TiDB server manipulates the Region leader data according to its own cache information, the Region leader has migrated, or the current TiKV Region information and the routing information of the TiDB cache are inconsistent. Generally, in this case, the TiDB server will automatically retrieve the latest routing data from PD and redo the previous operation.

12.3.4.3.8 What's the maximum number of concurrent connections that TiDB supports?

By default, there is no limit on the maximum number of connections per TiDB server. If too large concurrency leads to an increase of response time, it is recommended to increase the capacity by adding TiDB nodes.

12.3.4.3.9 How to view the creation time of a table?

The `create_time` of tables in the `information_schema` is the creation time.

12.3.4.3.10 What is the meaning of EXPENSIVE_QUERY in the TiDB log?

When TiDB is executing a SQL statement, the query will be `EXPENSIVE_QUERY` if each operator is estimated to process over 10000 pieces of data. You can modify the `tidb-server` configuration parameter to adjust the threshold and then restart the `tidb-server`.

12.3.4.4 TiKV server management

12.3.4.4.1 What is the recommended number of replicas in the TiKV cluster? Is it better to keep the minimum number for high availability?

3 replicas for each Region is sufficient for a testing environment. However, you should never operate a TiKV cluster with under 3 nodes in a production scenario. Depending on infrastructure, workload, and resiliency needs, you may wish to increase this number. It is worth noting that the higher the copy, the lower the performance, but the higher the security.

12.3.4.4.2 The cluster ID mismatch message is displayed when starting TiKV

This is because the cluster ID stored in local TiKV is different from the cluster ID specified by PD. When a new PD cluster is deployed, PD generates random cluster IDs. TiKV gets the cluster ID from PD and stores the cluster ID locally when it is initialized. The next time when TiKV is started, it checks the local cluster ID with the cluster ID in PD.

If the cluster IDs don't match, the `cluster ID mismatch` message is displayed and TiKV exits.

If you previously deploy a PD cluster, but then you remove the PD data and deploy a new PD cluster, this error occurs because TiKV uses the old data to connect to the new PD cluster.

12.3.4.4.3 The duplicated store address message is displayed when starting TiKV

This is because the address in the startup parameter has been registered in the PD cluster by other TiKVs. Common conditions that cause this error: There is no data folder in the path specified by TiKV `--data-dir` (no update `-data-dir` after deleting or moving), restart the TiKV with the previous parameters. Please try `store delete` function of `pd-ctl`, delete the previous store, and then restart TiKV.

12.3.4.4.4 TiKV primary node and secondary node use the same compression algorithm, why the results are different?

Currently, some files of TiKV primary node have a higher compression rate, which depends on the underlying data distribution and RocksDB implementation. It is normal that the data size fluctuates occasionally. The underlying storage engine adjusts data as needed.

12.3.4.4.5 What are the features of TiKV block cache?

TiKV implements the Column Family (CF) feature of RocksDB. By default, the KV data is eventually stored in the 3 CFs (default, write and lock) within RocksDB.

- The default CF stores real data and the corresponding parameter is in `[rocksdb.defaultcf]`.
- The write CF stores the data version information (MVCC) and index-related data, and the corresponding parameter is in `[rocksdb.writecf]`.
- The lock CF stores the lock information and the system uses the default parameter.
- The Raft RocksDB instance stores Raft logs. The default CF mainly stores Raft logs and the corresponding parameter is in `[raftdb.defaultcf]`.
- All CFs have a shared block-cache to cache data blocks and improve RocksDB read speed. The size of block-cache is controlled by the `block-cache-size` parameter. A larger value of the parameter means more hot data can be cached and is more favorable to read operation. At the same time, it consumes more system memory.
- Each CF has an individual write-buffer and the size is controlled by the `write-buffer-size` parameter.

12.3.4.4.6 Why is the TiKV channel full?

- The Raftstore thread is too slow or blocked by I/O. You can view the CPU usage status of Raftstore.
- TiKV is too busy (CPU, disk I/O, etc.) and cannot manage to handle it.

12.3.4.4.7 Why does TiKV frequently switch Region leader?

- Network problem results in the communication stuck among nodes. You can check Report failures monitoring.
- The node of the original main Leader is stuck, resulting in failure to reach out to the Follower in time.
- Raftstore thread stuck.

12.3.4.4.8 If a node is down, will the service be affected? If yes, how long?

TiKV uses Raft to replicate data among multiple replicas (by default 3 replicas for each Region). If one replica goes wrong, the other replicas can guarantee data safety. Based on the Raft protocol, if a single leader fails as the node goes down, a follower in another node is soon elected as the Region leader after a maximum of 2 * lease time (lease time is 10 seconds).

12.3.4.4.9 What are the TiKV scenarios that take up high I/O, memory, CPU, and exceed the parameter configuration?

Writing or reading a large volume of data in TiKV takes up high I/O, memory and CPU. Executing very complex queries costs a lot of memory and CPU resources, such as the scenario that generates large intermediate result sets.

12.3.4.4.10 Does TiKV support SAS/SATA disks or mixed deployment of SSD/SAS disks?

No. For OLTP scenarios, TiDB requires high I/O disks for data access and operation. As a distributed database with strong consistency, TiDB has some write amplification such as replica replication and bottom layer storage compaction. Therefore, it is recommended to use NVMe SSD as the storage disks in TiDB best practices. Mixed deployment of TiKV and PD is not supported.

12.3.4.4.11 Is the Range of the Key data table divided before data access?

No. It differs from the table splitting rules of MySQL. In TiKV, the table Range is dynamically split based on the size of Region.

12.3.4.4.12 How does Region split?

Region is not divided in advance, but it follows a Region split mechanism. When the Region size exceeds the value of the `region-max-size` or `region-max-keys` parameters, split is triggered. After the split, the information is reported to PD.

12.3.4.4.13 Does TiKV have the `innodb_flush_log_trx_commit` parameter like MySQL, to guarantee the security of data?

Yes. Currently, the standalone storage engine uses two RocksDB instances. One instance is used to store the raft-log. When the `sync-log` parameter in TiKV is set to true, each commit is mandatorily flushed to the raft-log. If a crash occurs, you can restore the KV data using the raft-log.

12.3.4.4.14 What is the recommended server configuration for WAL storage, such as SSD, RAID level, cache strategy of RAID card, NUMA configuration, file system, I/O scheduling strategy of the operating system?

WAL belongs to ordered writing, and currently, we do not apply a unique configuration to it. Recommended configuration is as follows:

- SSD
- RAID 10 preferred
- Cache strategy of RAID card and I/O scheduling strategy of the operating system: currently no specific best practices; you can use the default configuration in Linux 7 or later
- NUMA: no specific suggestion; for memory allocation strategy, you can use `interleave ↪ = all`
- File system: ext4

12.3.4.4.15 How is the write performance in the most strict data available mode (`sync-log = true`)?

Generally, enabling `sync-log` reduces about 30% of the performance. For write performance when `sync-log` is set to `false`, see [Performance test result for TiDB using Sysbench](#).

12.3.4.4.16 Can Raft + multiple replicas in the TiKV architecture achieve absolute data safety? Is it necessary to apply the most strict mode (`sync-log = true`) to a standalone storage?

Data is redundantly replicated between TiKV nodes using the [Raft Consensus Algorithm](#) to ensure recoverability should a node failure occur. Only when the data has been written into more than 50% of the replicas will the application return ACK (two out of three nodes). However, theoretically, two nodes might crash. Therefore, except for scenarios with less strict requirement on data safety but extreme requirement on performance, it is strongly recommended that you enable the `sync-log` mode.

As an alternative to using `sync-log`, you may also consider having five replicas instead of three in your Raft group. This would allow for the failure of two replicas, while still providing data safety.

For a standalone TiKV node, it is still recommended to enable the `sync-log` mode. Otherwise, the last write might be lost in case of a node failure.

12.3.4.4.17 Since TiKV uses the Raft protocol, multiple network roundtrips occur during data writing. What is the actual write delay?

Theoretically, TiDB has a write delay of 4 more network roundtrips than standalone databases.

12.3.4.4.18 Does TiDB have an InnoDB memcached plugin like MySQL which can directly use the KV interface and does not need the independent cache?

TiKV supports calling the interface separately. Theoretically, you can take an instance as the cache. Because TiDB is a distributed relational database, we do not support TiKV separately.

12.3.4.4.19 What is the Coprocessor component used for?

- Reduce the data transmission between TiDB and TiKV
- Make full use of the distributed computing resources of TiKV to execute computing pushdown.

12.3.4.4.20 The error message `IO error: No space left on device While appending to file` is displayed

This is because the disk space is not enough. You need to add nodes or enlarge the disk space.

12.3.4.4.21 Why does the OOM (Out of Memory) error occur frequently in TiKV?

The memory usage of TiKV mainly comes from the block-cache of RocksDB, which is 40% of the system memory size by default. When the OOM error occurs frequently in TiKV, you should check whether the value of `block-cache-size` is set too high. In addition, when multiple TiKV instances are deployed on a single machine, you need to explicitly configure the parameter to prevent multiple instances from using too much system memory that results in the OOM error.

12.3.4.4.22 Can both TiDB data and RawKV data be stored in the same TiKV cluster?

No. TiDB (or data created from the transactional API) relies on a specific key format. It is not compatible with data created from RawKV API (or data from other RawKV-based services).

12.3.4.5 TiDB testing

12.3.4.5.1 What is the performance test result for TiDB using Sysbench?

At the beginning, many users tend to do a benchmark test or a comparison test between TiDB and MySQL. We have also done a similar official test and find the test result is consistent at large, although the test data has some bias. Because the architecture of TiDB differs greatly from MySQL, it is hard to find a benchmark point. The suggestions are as follows:

- Do not spend too much time on the benchmark test. Pay more attention to the difference of scenarios using TiDB.
- See [Performance test result for TiDB using Sysbench](#).

12.3.4.5.2 What's the relationship between the TiDB cluster capacity (QPS) and the number of nodes? How does TiDB compare to MySQL?

- Within 10 nodes, the relationship between TiDB write capacity (Insert TPS) and the number of nodes is roughly 40% linear increase. Because MySQL uses single-node write, its write capacity cannot be scaled.
- In MySQL, the read capacity can be increased by adding secondary database, but the write capacity cannot be increased except using sharding, which has many problems.
- In TiDB, both the read and write capacity can be easily increased by adding more nodes.

12.3.4.5.3 The performance test of MySQL and TiDB by our DBA shows that the performance of a standalone TiDB is not as good as MySQL

TiDB is designed for scenarios where sharding is used because the capacity of a MySQL standalone is limited, and where strong consistency and complete distributed transactions are required. One of the advantages of TiDB is pushing down computing to the storage nodes to execute concurrent computing.

TiDB is not suitable for tables of small size (such as below ten million level), because its strength in concurrency cannot be shown with a small size of data and limited Regions. A typical example is the counter table, in which records of a few lines are updated high frequently. In TiDB, these lines become several Key-Value pairs in the storage engine, and then settle into a Region located on a single node. The overhead of background replication to guarantee strong consistency and operations from TiDB to TiKV leads to a poorer performance than a MySQL standalone.

12.3.4.6 Backup and restoration

12.3.4.6.1 How to back up data in TiDB?

Currently, for the backup of a large volume of data, the preferred method is using [BR](#). Otherwise, the recommended tool is [Dumpling](#). Although the official MySQL tool `mysqldump` is also supported in TiDB to back up and restore data, its performance is worse than [BR](#) and it needs much more time to back up and restore large volumes of data.

12.3.5 Monitoring

- For details of Prometheus monitoring framework, see [Overview of the Monitoring Framework](#).
- For details of key metrics of monitoring, see [Key Metrics](#).

12.3.5.1 Is there a better way of monitoring the key metrics?

The monitoring system of TiDB consists of Prometheus and Grafana. From the dashboard in Grafana, you can monitor various running metrics of TiDB which include the monitoring metrics of system resources, of client connection and SQL operation, of internal communication and Region scheduling. With these metrics, the database administrator can better understand the system running status, running bottlenecks and so on. In the practice of monitoring these metrics, we list the key metrics of each TiDB component. Generally you only need to pay attention to these common metrics. For details, see [official documentation](#).

12.3.5.2 The Prometheus monitoring data is deleted every 15 days by default. Could I set it to two months or delete the monitoring data manually?

Yes. Find the startup script on the machine where Prometheus is started, edit the startup parameter and restart Prometheus.

```
--storage.tsdb.retention="60d"
```

12.3.5.3 Region Health monitor

In TiDB 2.0, Region health is monitored in the PD metric monitoring page, in which the Region Health monitoring item shows the statistics of all the Region replica status. `miss` means shortage of replicas and `extra` means the extra replica exists. In addition, `Region → Health` also shows the isolation level by `label`. `level-1` means the Region replicas are isolated physically in the first `label` level. All the Regions are in `level-0` when `location → label` is not configured.

12.3.5.4 What is the meaning of `selectsimplefull` in Statement Count monitor?

It means full table scan but the table might be a small system table.

12.3.5.5 What is the difference between QPS and Statement OPS in the monitor?

The QPS statistics is about all the SQL statements, including `use database`, `load data`, `begin`, `commit`, `set`, `show`, `insert` and `select`.

The Statement OPS statistics is only about applications related SQL statements, including `select`, `update` and `insert`, therefore the Statement OPS statistics matches the applications better.

12.4 Upgrade and After Upgrade FAQs

This document introduces some FAQs and their solutions when or after you upgrade TiDB.

12.4.1 Upgrade FAQs

This section lists some FAQs and their solutions when you upgrade TiDB.

12.4.1.1 What are the effects of rolling updates?

When you apply rolling updates to the TiDB services, the running application is not affected. You need to configure the minimum cluster topology (TiDB * 2, PD * 3, TiKV * 3). If the Pump or Drainer service is involved in the cluster, it is recommended to stop Drainer before rolling updates. When you upgrade TiDB, Pump is also upgraded.

12.4.1.2 How to upgrade TiDB using the binary?

It is not recommended to deploy TiDB using the binary. The TiDB support for upgrading using Binary is not as friendly as using TiUP. It is recommended to [deploy TiDB using TiUP](#).

12.4.2 After upgrade FAQs

This section lists some FAQs and their solutions after you upgrade TiDB.

12.4.2.1 The character set (charset) errors when executing DDL operations

In v2.1.0 and earlier versions (including all versions of v2.0), the character set of TiDB is UTF-8 by default. But starting from v2.1.1, the default character set has been changed into UTF8MB4.

If you explicitly specify the charset of a newly created table as UTF-8 in v2.1.0 or earlier versions, then you might fail to execute DDL operations after upgrading TiDB to v2.1.1.

To avoid this issue, you need to pay attention to:

- Before v2.1.3, TiDB does not support modifying the charset of the column. Therefore, when you execute DDL operations, you need to make sure that the charset of the new column is consistent with that of the original column.
- Before v2.1.3, even if the charset of the column is different from that of the table, `show create table` does not show the charset of the column. But as shown in the following example, you can view it by obtaining the metadata of the table through the HTTP API.

12.4.2.1.1 unsupported modify column charset utf8mb4 not match origin utf8

- Before upgrading, the following operations are executed in v2.1.0 and earlier versions.

```
create table t(a varchar(10)) charset=utf8;
```

```
Query OK, 0 rows affected
Time: 0.106s
```

```
show create table t;
```

```
+-----+-----+
| Table | Create Table           |
+-----+-----+
| t     | CREATE TABLE `t` (
|       |   `a` varchar(10) DEFAULT NULL
|       | ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin |
+-----+-----+
1 row in set
Time: 0.006s
```

- After upgrading, the following error is reported in v2.1.1 and v2.1.2 but there is no such error in v2.1.3 and the later versions.

```
alter table t change column a a varchar(20);
```

```
ERROR 1105 (HY000): unsupported modify column charset utf8mb4 not match
→ origin utf8
```

Solution:

You can explicitly specify the column charset as the same with the original charset.

```
alter table t change column a a varchar(22) character set utf8;
```

- According to Point #1, if you do not specify the column charset, UTF8MB4 is used by default, so you need to specify the column charset to make it consistent with the original one.
- According to Point #2, you can obtain the metadata of the table through the HTTP API, and find the column charset by searching the column name and the keyword “Charset”.

```
curl "http://$IP:10080/schema/test/t" | python -m json.tool
```

A python tool is used here to format JSON, which is not required and only for the convenience to add the comments.

```
{
    "ShardRowIDBits": 0,
    "auto_inc_id": 0,
    "charset": "utf8", # The charset of the table.
    "collate": "",
    "cols": [           # The relevant information about the columns.
        {
            ...
            "id": 1,
            "name": {
                "L": "a",
                "O": "a" # The column name.
            },
            "offset": 0,
            "origin_default": null,
            "state": 5,
            "type": {
                "Charset": "utf8", # The charset of column a.
                "Collate": "utf8_bin",
                "Decimal": 0,
                "Elems": null,
                "Flag": 0,
                "Flen": 10,
                "Tp": 15
            }
        }
    ],
    ...
}
```

12.4.2.1.2 unsupported modify charset from utf8mb4 to utf8

- Before upgrading, the following operations are executed in v2.1.1 and v2.1.2.

```
create table t(a varchar(10)) charset=utf8;
```

```
Query OK, 0 rows affected
Time: 0.109s
```

```
show create table t;
```

```
+-----+-----+
| Table | Create Table |
+-----+-----+
| t     | CREATE TABLE `t` (
|       |   `a` varchar(10) DEFAULT NULL
|       | ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin |
+-----+-----+
```

In the above example, `show create table` only shows the charset of the table, but the charset of the column is actually UTF8MB4, which can be confirmed by obtaining the schema through the HTTP API. However, when a new table is created, the charset of the column should stay consistent with that of the table. This bug has been fixed in v2.1.3.

- After upgrading, the following operations are executed in v2.1.3 and the later versions.

```
show create table t;
```

```
+-----+-----+
←
| Table | Create Table |
+-----+-----+
←
| t     | CREATE TABLE `t` (
|       |   `a` varchar(10) CHARSET utf8mb4 COLLATE utf8mb4_bin DEFAULT
|       | → NULL |
|       | ) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_bin |
+-----+-----+
←
1 row in set
Time: 0.007s
```

```
alter table t change column a a varchar(20);
```

```
ERROR 1105 (HY000): unsupported modify charset from utf8mb4 to utf8
```

Solution:

- Starting from v2.1.3, TiDB supports modifying the charsets of the column and the table, so it is recommended to modify the table charset into UTF8MB4.

```
alter table t convert to character set utf8mb4;
```

- You can also specify the column charset as done in Issue #1, making it stay consistent with the original column charset (UTF8MB4).

```
alter table t change column a a varchar(20) character set utf8mb4;
```

12.4.2.1.3 ERROR 1366 (HY000): incorrect utf8 value f09f8c80(□) for column a

In TiDB v2.1.1 and earlier versions, if the charset is UTF-8, there is no UTF-8 Unicode encoding check on the inserted 4-byte data. But in v2.1.2 and the later versions, this check is added.

- Before upgrading, the following operations are executed in v2.1.1 and earlier versions.

```
create table t(a varchar(100) charset utf8);
```

```
Query OK, 0 rows affected
```

```
insert t values (unhex('f09f8c80'));
```

```
Query OK, 1 row affected
```

- After upgrading, the following error is reported in v2.1.2 and the later versions.

```
insert t values (unhex('f09f8c80'));
```

```
ERROR 1366 (HY000): incorrect utf8 value f09f8c80(□) for column a
```

Solution:

- In v2.1.2: this version does not support modifying the column charset, so you have to skip the UTF-8 check.

```
set @@session.tidb_skip_utf8_check=1;
```

```
Query OK, 0 rows affected
```

```
insert t values (unhex('f09f8c80'));
```

```
Query OK, 1 row affected
```

- In v2.1.3 and the later versions: it is recommended to modify the column charset into UTF8MB4. Or you can set `tidb_skip_utf8_check` to skip the UTF-8 check. But if you skip the check, you might fail to replicate data from TiDB to MySQL because MySQL executes the check.

```
alter table t change column a a varchar(100) character set utf8mb4;
```

```
Query OK, 0 rows affected
```

```
insert t values (unhex('f09f8c80'));
```

```
Query OK, 1 row affected
```

Specifically, you can use the variable `tidb_skip_utf8_check` to skip the legal UTF-8 and UTF8MB4 check on the data. But if you skip the check, you might fail to replicate the data from TiDB to MySQL because MySQL executes the check.

If you only want to skip the UTF-8 check, you can set `tidb_check_mb4_value_in_utf8` . This variable is added to the `config.toml` file in v2.1.3, and you can modify `check-mb4-value-in-utf8` in the configuration file and then restart the cluster to enable it.

Starting from v2.1.5, you can set `tidb_check_mb4_value_in_utf8` through the HTTP API and the session variable:

- HTTP API (the HTTP API can be enabled only on a single server)

- * To enable HTTP API:

```
curl -X POST -d "check_mb4_value_in_utf8=1" http://{TiDBIP
    ↪ }:10080/settings
```

- * To disable HTTP API:

```
curl -X POST -d "check_mb4_value_in_utf8=0" http://{TiDBIP
    ↪ }:10080/settings
```

- Session variable

- * To enable session variable:

```
set @@session.tidb_check_mb4_value_in_utf8 = 1;
```

- * To disable session variable:

```
set @@session.tidb_check_mb4_value_in_utf8 = 0;
```

12.5 High Availability FAQs

This document summarizes the FAQs related to high availability of TiDB.

12.5.1 How is TiDB strongly consistent?

Data is redundantly copied between TiKV nodes using the [Raft consensus algorithm](#) to ensure recoverability should a node failure occur.

At the bottom layer, TiKV uses a model of replication log + State Machine to replicate data. For the write requests, the data is written to a Leader and the Leader then replicates the command to its Followers in the form of log. When the majority of nodes in the cluster receive this log, this log is committed and can be applied into the State Machine.

12.5.2 What's the recommended solution for the deployment of three geo-distributed data centers?

The architecture of TiDB guarantees that it fully supports geo-distribution and multi-activeness. Your data and applications are always-on. All the outages are transparent to your applications and your data can recover automatically. The operation depends on the network latency and stability. It is recommended to keep the latency within 5ms. Currently, we already have similar use cases. For details, contact info@pingcap.com.

12.6 High Reliability FAQs

This document summarizes the FAQs related to high reliability of TiDB.

12.6.1 Does TiDB support modifying the MySQL version string of the server to a specific one that is required by the security vulnerability scanning tool?

Since v3.0.8, TiDB supports modifying the version string of the server by modifying `server-version` in the configuration file. When you deploy TiDB using TiUP, you can also specify the proper version string by executing `tiup cluster edit-config <cluster-name>`:

```
server_configs:  
  tidb:  
    server-version: 'YOUR_VERSION_STRING'
```

Use the `tiup cluster reload <cluster-name> -R tidb` command to make the modification above take effect to avoid the failure of security vulnerability scan.

12.6.2 What authentication protocols does TiDB support? What's the process?

Like MySQL, TiDB supports the SASL protocol for user login authentication and password processing.

When the client connects to TiDB, the challenge-response authentication mode starts. The process is as follows:

1. The client connects to the server.
2. The server sends a random string challenge to the client.
3. The client sends the username and response to the server.
4. The server verifies the response.

12.6.3 How to modify the user password and privilege?

To modify the user password in TiDB, it is recommended to use `set password for '→ root'@'%' = '0101001';` or `alter`, not `update mysql.user` which might lead to the condition that the password in other nodes is not refreshed timely.

It is recommended to use the official standard statements when modifying the user password and privilege. For details, see [TiDB user account management](#).

12.7 Migration FAQs

This document summarizes the frequently asked questions (FAQs) related to TiDB data migration.

For the frequently asked questions about migration-related tools, click the corresponding links in the list below:

- [Backup & Restore FAQ](#)
- [TiDB Binlog FAQ](#)
- [TiDB Lightning FAQs](#)
- [TiDB Data Migration \(DM\) FAQs](#)
- [Troubleshoot TiCDC](#)

12.7.1 Full data export and import

12.7.1.1 How to migrate an application running on MySQL to TiDB?

Because TiDB supports most MySQL syntax, generally you can migrate your applications to TiDB without changing a single line of code in most cases.

12.7.1.2 Data import and export is slow, and many retries and EOF errors appear in the log of each component without other errors

If no other logical errors occur, retries and EOF errors might be caused by network issues. It is recommended to first use tools to check the network connectivity. In the following example, `iperf` is used for troubleshooting:

- Execute the following command on the server-side node where the retries and EOF errors occur:

```
iperf3 -s
```

- Execute the following command on the client-side node where the retries and EOF errors occur:

```
iperf3 -c <server-IP>
```

The following example is the output of a client node with a good network connection:

```
$ iperf3 -c 192.168.196.58
Connecting to host 192.168.196.58, port 5201
[ 5] local 192.168.196.150 port 55397 connected to 192.168.196.58 port 5201
[ ID] Interval      Transfer     Bitrate
[ 5]  0.00-1.00  sec  18.0 MBytes  150 Mbits/sec
[ 5]  1.00-2.00  sec  20.8 MBytes  175 Mbits/sec
[ 5]  2.00-3.00  sec  18.2 MBytes  153 Mbits/sec
[ 5]  3.00-4.00  sec  22.5 MBytes  188 Mbits/sec
[ 5]  4.00-5.00  sec  22.4 MBytes  188 Mbits/sec
[ 5]  5.00-6.00  sec  22.8 MBytes  191 Mbits/sec
[ 5]  6.00-7.00  sec  20.8 MBytes  174 Mbits/sec
[ 5]  7.00-8.00  sec  20.1 MBytes  168 Mbits/sec
[ 5]  8.00-9.00  sec  20.8 MBytes  175 Mbits/sec
[ 5]  9.00-10.00 sec  21.8 MBytes  183 Mbits/sec
- - - - -
[ ID] Interval      Transfer     Bitrate
[ 5]  0.00-10.00 sec  208 MBytes  175 Mbits/sec
[ 5]  0.00-10.00 sec  208 MBytes  174 Mbits/sec
                                         sender
                                         receiver

iperf Done.
```

If the output shows low network bandwidth and high bandwidth fluctuations, a large number of retries and EOF errors might appear in each component log. In this case, you need to consult your network service provider to improve the network quality.

If the output of each metric looks good, try to update each component. If the problem persists after the updating, you can [contact us](#).

12.7.1.3 If I accidentally import the MySQL user table into TiDB, or forget the password and cannot log in, how to deal with it?

Restart the TiDB service, add the `-skip-grant-table=true` parameter in the configuration file. Log into the cluster without password and recreate the user, or recreate the `mysql.user` table. For the specific table schema, search the official documentation.

12.7.1.4 How to export the data in TiDB?

You can use the following methods to export the data in TiDB:

- See MySQL uses mysqldump to export part of the table data in Chinese and export data using mysqldump and the WHERE clause.
- Use the MySQL client to export the results of select to a file.

12.7.1.5 How to migrate from DB2 or Oracle to TiDB?

To migrate all the data or migrate incrementally from DB2 or Oracle to TiDB, see the following solution:

- Use the official migration tool of Oracle, such as OGG, Gateway, CDC (Change Data Capture).
- Develop a program for importing and exporting data.
- Export Spool as text file, and import data using Load infile.
- Use a third-party data migration tool.

Currently, it is recommended to use OGG.

12.7.1.6 Error: java.sql.BatchUpdateException:statement count 5001 exceeds the transaction limitation while using Sqoop to write data into TiDB in batches

In Sqoop, --batch means committing 100 statements in each batch, but by default each statement contains 100 SQL statements. So, $100 * 100 = 10000$ SQL statements, which exceeds 5000, the maximum number of statements allowed in a single TiDB transaction.

Two solutions:

- Add the -Dsqoop.export.records.per.statement=10 option as follows:

```
sqoop export \
  -Dsqoop.export.records.per.statement=10 \
  --connect jdbc:mysql://mysql.example.com/sqoop \
  --username sqoop ${user} \
  --password ${passwd} \
  --table ${tab_name} \
  --export-dir ${dir} \
  --batch
```

- You can also increase the limited number of statements in a single TiDB transaction, but this will consume more memory.

12.7.1.7 Why does Dumpling return The local disk space is insufficient error or cause the upstream database to run out of memory when exporting a table?

This issue might have the following causes:

- The database's primary keys are not evenly distributed (for example, when you enable `SHARD_ROW_ID_BITS`).
- The upstream database is TiDB and the exported table is a partitioned table.

For the above cases, Dumpling splits excessively large data chunk for the export and sends queries with excessively large results. To address the issue, you can [contact us](#) to get the nightly version of Dumpling.

12.7.1.8 Does TiDB have a function like the Flashback Query in Oracle? Does it support DDL?

Yes, it does. And it supports DDL as well. For details, see [how TiDB reads data from history versions](#).

12.7.2 Migrate the data online

12.7.2.1 Is there a current solution to replicating data from TiDB to other databases like HBase and Elasticsearch?

No. Currently, the data replication depends on the application itself.

12.7.3 Migrate the traffic

12.7.3.1 How to migrate the traffic quickly?

It is recommended to migrate application data from MySQL to TiDB using [TiDB Data Migration](#) tool. You can migrate the read and write traffic in batches by editing the network configuration as needed. Deploy a stable network LB (HAproxy, LVS, F5, DNS, etc.) on the upper layer, in order to implement seamless migration by directly editing the network configuration.

12.7.3.2 Is there a limit for the total write and read capacity in TiDB?

The total read capacity has no limit. You can increase the read capacity by adding more TiDB servers. Generally the write capacity has no limit as well. You can increase the write capacity by adding more TiKV nodes.

12.7.3.3 The error message `transaction too large` is displayed

Due to the limitation of the underlying storage engine, each key-value entry (one row) in TiDB should be no more than 6MB. You can adjust the `txn-entry-size-limit` configuration value up to 120MB.

Distributed transactions need two-phase commit and the bottom layer performs the Raft replication. If a transaction is very large, the commit process would be quite slow and the write conflict is more likely to occur. Moreover, the rollback of a failed transaction leads

to an unnecessary performance penalty. To avoid these problems, we limit the total size of key-value entries to no more than 100MB in a transaction by default. If you need larger transactions, modify the value of `txn-total-size-limit` in the TiDB configuration file. The maximum value of this configuration item is up to 10G. The actual limitation is also affected by the physical memory of the machine.

There are [similar limits](#) on Google Cloud Spanner.

12.7.3.4 How to import data in batches?

When you import data, insert in batches and keep the number of rows within 10,000 for each batch.

12.7.3.5 Does TiDB release space immediately after deleting data?

None of the `DELETE`, `TRUNCATE` and `DROP` operations release data immediately. For the `TRUNCATE` and `DROP` operations, after the TiDB GC (Garbage Collection) time (10 minutes by default), the data is deleted and the space is released. For the `DELETE` operation, the data is deleted but the space is not released according to TiDB GC. When subsequent data is written into RocksDB and executes `COMPACT`, the space is reused.

12.7.3.6 Can I execute DDL operations on the target table when loading data?

No. None of the DDL operations can be executed on the target table when you load data, otherwise the data fails to be loaded.

12.7.3.7 Does TiDB support the `replace into` syntax?

Yes. But the `load data` does not support the `replace into` syntax.

12.7.3.8 Why does the query speed getting slow after deleting data?

Deleting a large amount of data leaves a lot of useless keys, affecting the query efficiency. Currently the Region Merge feature is in development, which is expected to solve this problem. For details, see the [deleting data section in TiDB Best Practices](#).

12.7.3.9 What is the most efficient way of deleting data?

When deleting a large amount of data, it is recommended to use `Delete from t where → xx limit 5000;`. It deletes through the loop and uses `Affected Rows == 0` as a condition to end the loop, so as not to exceed the limit of transaction size. With the prerequisite of meeting business filtering logic, it is recommended to add a strong filter index column or directly use the primary key to select the range, such as `id >= 5000*n+m and id < 5000*(→ n+1)+m`.

If the amount of data that needs to be deleted at a time is very large, this loop method will get slower and slower because each deletion traverses backward. After deleting the previous

data, lots of deleted flags remain for a short period (then all will be processed by Garbage Collection) and influence the following Delete statement. If possible, it is recommended to refine the Where condition. See [details in TiDB Best Practices](#).

12.7.3.10 How to improve the data loading speed in TiDB?

- The [TiDB Lightning](#) tool is developed for distributed data import. It should be noted that the data import process does not perform a complete transaction process for performance reasons. Therefore, the ACID constraint of the data being imported during the import process cannot be guaranteed. The ACID constraint of the imported data can only be guaranteed after the entire import process ends. Therefore, the applicable scenarios mainly include importing new data (such as a new table or a new index) or the full backup and restoring (truncate the original table and then import data).
- Data loading in TiDB is related to the status of disks and the whole cluster. When loading data, pay attention to metrics like the disk usage rate of the host, TiClient Error, Backoff, Thread CPU and so on. You can analyze the bottlenecks using these metrics.

13 Glossary

13.1 A

13.1.1 ACID

ACID refers to the four key properties of a transaction: atomicity, consistency, isolation, and durability. Each of these properties is described below.

- **Atomicity** means that either all the changes of an operation are performed, or none of them are. TiDB ensures the atomicity of the [Region](#) that stores the Primary Key to achieve the atomicity of transactions.
- **Consistency** means that transactions always bring the database from one consistent state to another. In TiDB, data consistency is ensured before writing data to the memory.
- **Isolation** means that a transaction in process is invisible to other transactions until it completes. This allows concurrent transactions to read and write data without sacrificing consistency. TiDB currently supports the isolation level of [REPEATABLE READ](#).
- **Durability** means that once a transaction is committed, it remains committed even in the event of a system failure. TiKV uses persistent storage to ensure durability.

13.2 L

13.2.1 leader/follower/learner

Leader/Follower/Learner each corresponds to a role in a Raft group of [peers](#). The leader services all client requests and replicates data to the followers. If the group leader fails, one of the followers will be elected as the new leader. Learners are non-voting followers that only serves in the process of replica addition.

13.3 O

13.3.1 Old value

The “original value” in the incremental change log output by TiCDC. You can specify whether the incremental change log output by TiCDC contains the “original value”.

13.3.2 Operator

An operator is a collection of actions that applies to a Region for scheduling purposes. Operators perform scheduling tasks such as “migrate the leader of Region 2 to Store 5” and “migrate replicas of Region 2 to Store 1, 4, 5”.

An operator can be computed and generated by a [scheduler](#), or created by an external API.

13.3.3 Operator step

An operator step is a step in the execution of an operator. An operator normally contains multiple Operator steps.

Currently, available steps generated by PD include:

- [TransferLeader](#): Transfers leadership to a specified member
- [AddPeer](#): Adds peers to a specified store
- [RemovePeer](#): Removes a peer of a Region
- [AddLearner](#): Adds learners to a specified store
- [PromoteLearner](#): Promotes a specified learner to a voting member
- [SplitRegion](#): Splits a specified Region into two

13.4 P

13.4.1 pending/down

“Pending” and “down” are two special states of a peer. Pending indicates that the Raft log of followers or learners is vastly different from that of leader. Followers in pending cannot

be elected as leader. “Down” refers to a state that a peer ceases to respond to leader for a long time, which usually means the corresponding node is down or isolated from the network.

13.5 R

13.5.1 Region/peer/Raft group

Region is the minimal piece of data storage in TiKV, each representing a range of data (96 MiB by default). Each Region has three replicas by default. A replica of a Region is called a peer. Multiple peers of the same Region replicate data via the Raft consensus algorithm, so peers are also members of a Raft instance. TiKV uses Multi-Raft to manage data. That is, for each Region, there is a corresponding, isolated Raft group.

13.5.2 Region split

Regions are generated as data writes increase. The process of splitting is called Region split.

The mechanism of Region split is to use one initial Region to cover the entire key space, and generate new Regions through splitting existing ones every time the size of the Region or the number of keys has reached a threshold.

13.5.3 restore

Restore is the reverse of the backup operation. It is the process of bringing back the system to an earlier state by retrieving data from a prepared backup.

13.6 S

13.6.1 scheduler

Schedulers are components in PD that generate scheduling tasks. Each scheduler in PD runs independently and serves different purposes. The commonly used schedulers are:

- **balance-leader-scheduler**: Balances the distribution of leaders
- **balance-region-scheduler**: Balances the distribution of peers
- **hot-region-scheduler**: Balances the distribution of hot Regions
- **evict-leader-{store-id}**: Evicts all leaders of a node (often used for rolling upgrades)

13.6.2 Store

A store refers to the storage node in the TiKV cluster (an instance of `tikv-server`). Each store has a corresponding TiKV instance.

13.7 T

13.7.1 TSO

Because TiKV is a distributed storage system, it requires a global timing service, Timestamp Oracle (TSO), to assign a monotonically increasing timestamp. In TiKV, such a feature is provided by PD, and in Google [Spanner](#), this feature is provided by multiple atomic clocks and GPS.

14 Release Notes

14.1 TiDB Release Notes

14.1.1 5.3

- [5.3.0](#)

14.1.2 5.2

- [5.2.3](#)
- [5.2.2](#)
- [5.2.1](#)
- [5.2.0](#)

14.1.3 5.1

- [5.1.3](#)
- [5.1.2](#)
- [5.1.1](#)
- [5.1.0](#)

14.1.4 5.0

- [5.0.5](#)
- [5.0.4](#)
- [5.0.3](#)
- [5.0.2](#)
- [5.0.1](#)
- [5.0 GA](#)
- [5.0.0-rc](#)

14.1.5 4.0

- 4.0.16
- 4.0.15
- 4.0.14
- 4.0.13
- 4.0.12
- 4.0.11
- 4.0.10
- 4.0.9
- 4.0.8
- 4.0.7
- 4.0.6
- 4.0.5
- 4.0.4
- 4.0.3
- 4.0.2
- 4.0.1
- 4.0 GA
- 4.0.0-rc.2
- 4.0.0-rc.1
- 4.0.0-rc
- 4.0.0-beta.2
- 4.0.0-beta.1
- 4.0.0-beta

14.1.6 3.1

- 3.1.2
- 3.1.1
- 3.1.0 GA
- 3.1.0-rc
- 3.1.0-beta.2
- 3.1.0-beta.1
- 3.1.0-beta

14.1.7 3.0

- 3.0.20
- 3.0.19
- 3.0.18
- 3.0.17
- 3.0.16
- 3.0.15

- 3.0.14
- 3.0.13
- 3.0.12
- 3.0.11
- 3.0.10
- 3.0.9
- 3.0.8
- 3.0.7
- 3.0.6
- 3.0.5
- 3.0.4
- 3.0.3
- 3.0.2
- 3.0.1
- 3.0 GA
- 3.0.0-rc.3
- 3.0.0-rc.2
- 3.0.0-rc.1
- 3.0.0-beta.1
- 3.0.0-beta

14.1.8 2.1

- 2.1.19
- 2.1.18
- 2.1.17
- 2.1.16
- 2.1.15
- 2.1.14
- 2.1.13
- 2.1.12
- 2.1.11
- 2.1.10
- 2.1.9
- 2.1.8
- 2.1.7
- 2.1.6
- 2.1.5
- 2.1.4
- 2.1.3
- 2.1.2
- 2.1.1
- 2.1 GA
- 2.1 RC5

- 2.1 RC4
- 2.1 RC3
- 2.1 RC2
- 2.1 RC1
- 2.1 Beta

14.1.9 2.0

- 2.0.11
- 2.0.10
- 2.0.9
- 2.0.8
- 2.0.7
- 2.0.6
- 2.0.5
- 2.0.4
- 2.0.3
- 2.0.2
- 2.0.1
- 2.0 GA
- 2.0 RC5
- 2.0 RC4
- 2.0 RC3
- 2.0 RC1
- 1.1 Beta
- 1.1 Alpha

14.1.10 1.0

- 1.0.8
- 1.0.7
- 1.0.6
- 1.0.5
- 1.0.4
- 1.0.3
- 1.0.2
- 1.0.1
- 1.0 GA
- Pre-GA
- RC4
- RC3
- RC2
- RC1

14.2 TiDB Release Timeline

This document shows all the released TiDB versions in reverse chronological order.

Version	Release Date
4.0.16	2021-12-17
5.1.3	2021-12-03
5.0.5	2021-12-03
5.2.3	2021-12-03
5.3.0	2021-11-30
5.2.2	2021-10-29
5.1.2	2021-09-27
5.0.4	2021-09-27
4.0.15	2021-09-27
5.2.1	2021-09-09
5.2.0	2021-08-27
5.1.1	2021-07-30
4.0.14	2021-07-27
5.0.3	2021-07-02
5.1.0	2021-06-24
5.0.2	2021-06-10
4.0.13	2021-05-28
5.0.1	2021-04-24
5.0.0	2021-04-07
4.0.12	2021-04-02
4.0.11	2021-02-26
4.0.10	2021-01-15
5.0.0-rc	2021-01-12
3.0.20	2020-12-25
4.0.9	2020-12-21
4.0.8	2020-10-30
4.0.7	2020-09-29
3.0.19	2020-09-25
4.0.6	2020-09-15
4.0.5	2020-08-31
3.0.18	2020-08-21
3.0.17	2020-08-03
4.0.4	2020-07-31
4.0.3	2020-07-24
3.0.16	2020-07-03
4.0.2	2020-07-01
4.0.1	2020-06-12
3.0.15	2020-06-05
3.1.2	2020-06-04
4.0.0	2020-05-28

Version	Release Date
4.0.0-rc.2	2020-05-15
3.0.14	2020-05-09
3.1.1	2020-04-30
4.0.0-rc.1	2020-04-28
3.0.13	2020-04-22
3.1.0	2020-04-16
4.0.0-rc	2020-04-08
3.1.0-rc	2020-04-02
4.0.0-beta.2	2020-03-18
3.0.12	2020-03-16
3.1.0-beta.2	2020-03-09
3.0.11	2020-03-04
4.0.0-beta.1	2020-02-28
3.0.10	2020-02-20
4.0.0-beta	2020-01-17
3.0.9	2020-01-14
3.1.0-beta.1	2020-01-10
3.0.8	2019-12-31
2.1.19	2019-12-27
3.1.0-beta	2019-12-20
3.0.7	2019-12-04
3.0.6	2019-11-28
2.1.18	2019-11-04
3.0.5	2019-10-25
3.0.4	2019-10-08
2.1.17	2019-09-11
3.0.3	2019-08-29
2.1.16	2019-08-15
3.0.2	2019-08-07
2.1.15	2019-07-18
3.0.1	2019-07-16
2.1.14	2019-07-04
3.0.0	2019-06-28
3.0.0-rc.3	2019-06-21
2.1.13	2019-06-21
2.1.12	2019-06-13
2.1.11	2019-06-03
3.0.0-rc.2	2019-05-28
2.1.10	2019-05-22
3.0.0-rc.1	2019-05-10
2.1.9	2019-05-06
2.1.8	2019-04-12
2.1.7	2019-03-28

Version	Release Date
3.0.0-beta.1	2019-03-26
2.1.6	2019-03-15
2.1.5	2019-02-28
2.1.4	2019-02-15
2.1.3	2019-01-28
3.0.0-beta	2019-01-19
2.0.11	2019-01-03
2.1.2	2018-12-22
2.0.10	2018-12-18
2.1.1	2018-12-12
2.1.0	2018-11-30
2.0.9	2018-11-19
2.1.0-rc.5	2018-11-12
2.1.0-rc.4	2018-10-23
2.0.8	2018-10-16
2.1.0-rc.3	2018-09-29
2.1.0-rc.2	2018-09-14
2.0.7	2018-09-07
2.1.0-rc.1	2018-08-24
2.0.6	2018-08-06
2.0.5	2018-07-06
2.1.0-beta	2018-06-29
2.0.4	2018-06-15
2.0.3	2018-06-01
2.0.2	2018-05-21
2.0.1	2018-05-16
2.0.0	2018-04-27
2.0.0-rc.5	2018-04-17
2.0.0-rc.4	2018-03-30
2.0.0-rc.3	2018-03-23
2.0.0-rc.1	2018-03-09
1.1.0-beta	2018-02-24
1.0.8	2018-02-11
1.0.7	2018-01-22
1.1.0-alpha	2018-01-19
1.0.6	2018-01-08
1.0.5	2017-12-26
1.0.4	2017-12-11
1.0.3	2017-11-28
1.0.2	2017-11-13
1.0.1	2017-11-01
1.0.0	2017-10-16
Pre-GA	2017-08-30

Version	Release Date
rc4	2017-08-04
rc3	2017-06-16
rc2	2017-03-01
rc1	2016-12-23

14.3 v5.3

14.3.1 TiDB 5.3 Release Notes

Release date: November 30, 2021

TiDB version: 5.3.0

In v5.3, the key new features or improvements are as follows:

- Introduce temporary tables to simplify your application logic and improve performance
- Support setting attributes for tables and partitions
- Support creating users with the least privileges on TiDB Dashboard to enhance system security
- Optimize the timestamp processing flow in TiDB to improve the overall performance
- Enhance the performance of TiDB Data Migration (DM) so that data is migrated from MySQL to TiDB with lower latency
- Support parallel import using multiple TiDB Lightning instances to improve the efficiency of full data migration
- Support saving and restoring the on-site information of a cluster with a single SQL statement, which helps improve the efficiency of troubleshooting issues relating to execution plans
- Support the continuous profiling experimental feature to improve the observability of database performance
- Continue optimizing the storage and computing engines to improve the system performance and stability

14.3.1.1 Compatibility changes

Note:

When upgrading from an earlier TiDB version to v5.3.0, if you want to know the compatibility change notes of all intermediate versions, you can check the [Release Notes](#) of the corresponding version.

14.3.1.1.1 System variables

Variable name	Change type	Description
<code>tidb_enable_noop_functions</code>	Modified →	Temporary tables are now supported by TiDB so CREATE → TEMPORARY → TABLE and DROP → TEMPORARY → TABLE no longer require enabling <code>tidb_enable_noop_functions</code> → .

Variable		
name	Change type	Description
<code>tidb_enable_pseudo</code>	Newly added →	<p><code>control_state</code></p> <p>Controls the behavior of the optimizer when the statistics on a table expire. The default value is <code>ON</code>. When the number of modified rows in the table is greater than 80% of the total rows (This ratio can be adjusted by the configuration <code>pseudo-estimate</code> → <code>-ratio</code>), the optimizer considers that the statistics other than the total number of rows are no longer reliable and use pseudo statistics instead.</p> <p>When you set the value as <code>OFF</code>, even if the statistics expire, the optimizer still uses them.</p>

Variable name	Change type	Description
<code>tidb_enable_tso_follower_proxy</code>	Newly added proxy →	<p>Determines whether to enable or disable the TSO Follower Proxy feature. The default value is OFF, which means the TSO Follower Proxy feature is disabled.</p> <p>At this time, TiDB only gets TSO from PD leader. When this feature is enabled, TiDB evenly sends the requests to all PD nodes when acquiring TSO. The PD follower then forwards the TSO requests to reduce the CPU pressure of PD leader.</p>

Variable			
name	Change type	Description	
<code>tidb_tso_client_new_batched_max_set_time</code>	New by <code>tidb_tso_client_max_set_time</code>	maximum waiting time for a batch saving operation when TiDB requests TSO from PD.	The default value is 0, which means no additional waiting.
<code>tidb_tmp_table_new_max_size</code>	New by <code>tidb_tmp_table_max_size</code>	Limits the maximum size of a single temporary table. If the temporary table exceeds this size, an error will occur.	

14.3.1.1.2 Configuration file parameters

Configuration file	Configuration item	Change type	Description
TiDB	<code>prepared-plan-cache.capacity</code>	Modified	Controls the number of cached statements. The default value is changed from 100 to 1000.

file	item	Change type	Description
TiKV	<code>storage.</code> ↳ <code>reserve</code> ↳ <code>-space</code>	Modified	Controls space reserved for disk protection when TiKV is started. Starting from v5.3.0, 80% of the reserved space is used as the extra disk space required for operations and maintenance when the disk space is insufficient, and the other 20% is used to store temporary files.
TiKV	<code>memory-</code> ↳ <code>usage-</code> ↳ <code>limit</code>	Modified	This configuration item is new in TiDB v5.3.0 and its value is calculated based on storage.block-cache.capacity.

file	item	Change type	Description
TiKV	raftstore. ↵ store- ↵ io-pool ↵ -size	Newly added	The allowable number of threads that process Raft I/O tasks, which is the size of the StoreWriter thread pool. When you modify the size of this thread pool, refer to Performance tuning for TiKV thread pools .

file	item	Change type	Description
TiKV	raftstore. ↳ raft- ↳ write- ↳ size- ↳ limit	Newly added threshold at which Raft data is written into the disk. If the data size is larger than the value of this configuration item, the data is written to the disk. When the value of raftstore. ↳ store- ↳ io-pool ↳ -size is 0, this configuration item does not take effect.	Determines the threshold at which Raft data is written into the disk. If the data size is larger than the value of this configuration item, the data is written to the disk. When the value of raftstore. ↳ store- ↳ io-pool ↳ -size is 0, this configuration item does not take effect.

file	item	Change type	Description
TiKV	raftstore. ↳ raft- ↳ msg- ↳ flush- ↳ interval ↳	Newly added	<p>Determines the interval at which Raft messages are sent in batches.</p> <p>The Raft messages in batches are sent at every interval specified by this configuration item.</p> <p>When the value of raftstore. ↳ store- ↳ io-pool ↳ -size is 0, this configuration item does not take effect.</p>
TiKV	raftstore. ↳ raft- ↳ reject- ↳ transfer ↳ -leader ↳ - ↳ duration ↳	Deleted	<p>Determines the smallest duration that a Leader is transferred to a newly added node.</p>

Configuration file	Configuration item	Change type	Description
PD	<code>log.file.</code> ↳ <code>max-</code> ↳ <code>days</code>	Modified	Controls the maximum number of days that logs are retained for. The default value is changed from 1 to 0.
PD	<code>log.file.</code> ↳ <code>max-</code> ↳ <code>backups</code>	Modified	Controls the maximum number of logs that are retained for. The default value is changed from 7 to 0.

file	item	Change type	Description
PD	patrol- ↵ region- ↵ interval ↵	Modified	<p>Controls the running frequency at which replicaChecker checks the health state of a Region. The smaller this value is, the faster replicaChecker runs.</p> <p>Normally, you do not need to adjust this parameter. The default value is changed from 100ms to 10ms.</p>

file	item	Change type	Description
PD	max- ↵ snapshot ↵ -count	Modified	<p>Controls the maximum number of snapshots that a single store receives or sends at the same time.</p> <p>PD schedulers depend on this configuration to prevent the resources used for normal traffic from being preempted. The default value is changed from 3 to 64.</p>

file	item	Change type	Description
PD	max- ↵ pending ↵ -peer- ↵ count	Modified	Controls the maximum number of pending peers in a single store. PD schedulers depend on this configuration to prevent too many Regions with outdated logs from being generated on some nodes. The default value is changed from 16 to 64.

14.3.1.1.3 Others

- Temporary tables:
 - If you have created local temporary tables in a TiDB cluster earlier than v5.3.0, these tables are actually ordinary tables, and handled as ordinary tables after the cluster is upgraded to v5.3.0 or a later version. If you have created global temporary tables in a TiDB cluster of v5.3.0 or a later version, when the cluster is downgraded to a version earlier than v5.3.0, these tables are handled as ordinary tables and cause a data error.
 - Since v5.3.0, TiCDC and BR support [global temporary tables](#). If you use TiCDC and BR of a version earlier than v5.3.0 to replicate global temporary tables to the downstream, a table definition error occurs.
 - The following clusters are expected to be v5.3.0 or later; otherwise, data error is reported when you create a global temporary table:

- * the cluster to be imported using TiDB ecosystem tools
 - * the cluster restored using TiDB ecosystem tools
 - * the downstream cluster in a replication task using TiDB ecosystem tools
- For the compatibility information of temporary tables, refer to [Compatibility with MySQL temporary tables](#) and [Compatibility restrictions with other TiDB features](#).
- For releases earlier than v5.3.0, TiDB reports an error when a system variable is set to an illegal value. For v5.3.0 and later releases, TiDB returns success with a warning such as “|Warning | 1292 | Truncated incorrect xxx: ‘xx’” when a system variable is set to an illegal value.
 - Fix the issue that the `SHOW VIEW` permission is not required to execute `SHOW CREATE → VIEW`. Now you are expected to have the `SHOW VIEW` permission to execute the `SHOW CREATE VIEW` statement.
 - The system variable `sql_auto_is_null` is added to the noop functions. When `tidb_enable_noop_functions = 0/OFF`, modifying this variable value causes an error.
 - The `GRANT ALL ON performance_schema.*` syntax is no longer permitted. If you execute this statement in TiDB, an error occurs.
 - Fix the issue that auto-analyze is unexpectedly triggered outside the specified time period when new indexes are added before v5.3.0. In v5.3.0, after you set the time period through the `tidb_auto_analyze_start_time` and `tidb_auto_analyze_end_time` variables, auto-analyze is triggered only during this time period.
 - The default storage directory for plugins is changed from “” to `/data/deploy/plugin`.
 - The DM code is migrated to [the folder “dm” in TiCDC code repository](#). Now DM follows TiDB in version numbers. Next to v2.0.x, the new DM version is v5.3.0, and you can upgrade from v2.0.x to v5.3.0 without any risk.

14.3.1.2 New features

14.3.1.2.1 SQL

- **Use SQL interface to set placement rules for data (experimental feature)**

Support the `[CREATE | ALTER] PLACEMENT POLICY` syntax that provides a SQL interface to set placement rules for data. Using this feature, you can specify tables and partitions to be scheduled to specific regions, data centers, racks, hosts, or replica count rules. This meets your application demands for lower cost and higher flexibility. The typical user scenarios are as follows:

- Merge multiple databases of different applications to reduce the cost on database maintenance, and achieve application resource isolation through the rule configuration
- Increase replica count for important data to improve the application availability and data reliability
- Store new data into SSDs and store old data into HDDs to lower the cost on data archiving and storage
- Schedule the leaders of hotspot data to high-performance TiKV instances
- Separate cold data to lower-cost storage mediums to improve cost efficiency

[User document, #18030](#)

- **Temporary tables**

Support the `CREATE [GLOBAL] TEMPORARY TABLE` statement to create temporary tables. Using this feature, you can easily manage the temporary data generated in the calculation process of an application. Temporary data is stored in memory and you can use the `tidb_tmp_table_max_size` variable to limit the size of a temporary table. TiDB supports the following types of temporary tables:

- Global temporary tables
 - * Visible to all sessions in the cluster, and table schemas are persistent.
 - * Provides transaction-level data isolation. The temporary data is effective only in the transaction. After the transaction finishes, the data is automatically dropped.
- Local temporary tables
 - * Visible only to the current session, and tables schemas are not persistent.
 - * Supports duplicated table names. You do not need to design complicated naming rules for your application.
 - * Provides session-level data isolation, which enables you to design a simpler application logic. After the transaction finishes, the temporary tables are dropped.

[User document, #24169](#)

- **Support the FOR UPDATE OF TABLES syntax**

For a SQL statement that joins multiple tables, TiDB supports acquiring pessimistic locks on the rows correlated to the tables that are included in `OF TABLES`.

[User document, #28689](#)

- **Table attributes**

Support the `ALTER TABLE [PARTITION] ATTRIBUTES` statement that allows you to set attributes for a table or partition. Currently, TiDB only supports setting the `merge_option` attribute. By adding this attribute, you can explicitly control the Region merge behavior.

User scenarios: When you perform the `SPLIT TABLE` operation, if no data is inserted after a certain period of time, the empty Regions are automatically merged by default. In this case, you can set the table attribute to `merge_option=deny` to avoid the automatic merging of Regions.

[User document](#), #3839

14.3.1.2.2 Security

- **Support creating users with the least privileges on TiDB Dashboard**

The account system of TiDB Dashboard is consistent with that of TiDB SQL. Users accessing TiDB Dashboard are authenticated and authorized based on TiDB SQL users' privileges. Therefore, TiDB Dashboard requires limited privileges, or merely the read-only privilege. You can configure users to access TiDB Dashboard based on the principle of least privilege, thus avoiding access of high-privileged users.

It is recommended that you create a least-privileged SQL user to access and sign in with TiDB Dashboard. This avoids access of high-privileged users and improves security.

[User document](#)

14.3.1.2.3 Performance

- **Optimize the timestamp processing flow of PD**

TiDB optimizes its timestamp processing flow and reduces the timestamp processing load of PD by enabling PD Follower Proxy and modifying the batch waiting time required when the PD client requests TSO in batches. This helps improve the overall scalability of the system.

- Support enabling or disabling PD Follower Proxy through the system variable `tidb_enable_tso_follower_proxy`. Suppose that the TSO requests load of PD is too high. In this case, enabling PD follower proxy can batch forward the TSO requests collected during the request cycle on followers to the leader nodes. This solution can effectively reduce the number of direct interactions between clients and leaders, reduce the pressure of the load on leaders, and improve the overall performance of TiDB.

Note:

When the number of clients is small and the PD leader CPU load is not full, it is NOT recommended to enable PD Follower Proxy.

- Support using the system variable `tidb_tso_client_batch_max_wait_time` to set the maximum waiting time needed for the PD client to batch request TSO. The unit of this time is milliseconds. In case that PD has a high TSO requests load, you can reduce the load and improve the throughput by increasing the waiting time to get a larger batch size.

Note:

When the TSO request load is not high, it is NOT recommended to modify this variable value.

[User document](#), #3149

14.3.1.2.4 Stability

- **Support Online Unsafe Recovery after some stores are permanently damaged (experimental feature)**

Support the `unsafe remove-failed-stores` command that performs online data unsafe recovery. Suppose that the majority of data replicas encounter issues like permanent damage (such as disk damage), and these issues cause the data ranges in an application to be unreadable or unwritable. In this case, you can use the Online Unsafe Recovery feature implemented in PD to recover the data, so that the data is readable or writable again.

It is recommended to perform the feature-related operations with the support of the TiDB team.

[User document](#), #10483

14.3.1.2.5 Data migration

- **DM replication performance enhanced**

Supports the following features to ensure lower-latency data replication from MySQL to TiDB:

- Compact multiple updates on a single row into one statement
- Merge batch updates of multiple rows into one statement

- **Add DM OpenAPI to better maintain DM clusters (experimental feature)**

DM provides the OpenAPI feature for querying and operating the DM cluster. It is similar to the feature of `dmctl tools`.

Currently, DM OpenAPI is an experimental feature and disabled by default. It is not recommended to use it in a production environment.

[User document](#)

- **TiDB Lightning Parallel Import**

TiDB Lightning provides parallel import capability to extend the original feature. It allows you to deploy multiple Lightning instances at the same time to import single tables or multiple tables to downstream TiDB in parallel. Without changing the way customers use it, it greatly improves the data migration ability, allowing you to migrate data in a more real-time way to further process, integrate and analyze them. It improves the efficiency of enterprise data management.

In our test, using 10 TiDB Lightning instances, a total of 20 TiB MySQL data can be imported to TiDB within 8 hours. The performance of multiple table import is also improved. A single TiDB Lightning instance can support importing at 250 GiB/h, and the overall migration is 8 times faster than the original performance.

[User document](#)

- **TiDB Lightning Prechecks**

TiDB Lightning provides the ability to check the configuration before running a migration task. It is enabled by default. This feature automatically performs some routine checks for disk space and execution configuration. The main purpose is to ensure that the whole subsequent import process goes smoothly.

[User document](#)

- **TiDB Lightning supports importing files of GBK character set**

You can specify the character set of the source data file. TiDB Lightning will convert the source file from the specified character set to UTF-8 encoding during the import process.

[User document](#)

- **Sync-diff-inspector improvement**

- Improve the comparison speed from 375 MB/s to 700 MB/s
- Reduce the memory consumption of TiDB nodes by nearly half during comparison
- Optimize the user interface and display the progress bar during comparison

[User document](#)

14.3.1.2.6 Diagnostic efficiency

- **Save and restore the on-site information of a cluster**

When you locate and troubleshoot the issues of a TiDB cluster, you often need to provide information on the system and the query plan. To help you get the information and troubleshoot cluster issues in a more convenient and efficient way, the PLAN REPLAY command is introduced in TiDB v5.3.0. This command enables you to easily save and restore the on-site information of a cluster, improves the efficiency of troubleshooting, and helps you more easily archive the issues for management.

The features of PLAN REPLAYER are as follows:

- Exports the information of a TiDB cluster at an on-site troubleshooting to a ZIP-formatted file for storage.
- Imports into a cluster the ZIP-formatted file exported from another TiDB cluster. This file contains the information of the latter TiDB cluster at an on-site troubleshooting.

[User document](#), #26325

14.3.1.2.7 TiDB data share subscription

- **TiCDC Eventually Consistent Replication**

TiCDC provides the eventually consistent replication capability in disaster scenarios. When a disaster occurs in the primary TiDB cluster and the service cannot be resumed in a short period of time, TiCDC needs to provide the ability to ensure the consistency of data in the secondary cluster. Meanwhile, TiCDC needs to allow the business to quickly switch the traffic to the secondary cluster to avoid the database being unavailable for a long time and affecting the business.

This feature supports TiCDC to replicate incremental data from a TiDB cluster to the secondary relational database TiDB/Aurora/MySQL/MariaDB. In case the primary cluster crashes, TiCDC can recover the secondary cluster to a certain snapshot in the primary cluster within 5 minutes, given the condition that before disaster the replication status of TiCDC is normal and replication lag is small. It allows data loss of less than 30 minutes, that is, RTO \leq 5min, and RPO \leq 30min.

[User document](#)

- **TiCDC supports the HTTP protocol OpenAPI for managing TiCDC tasks**

Since TiDB v5.3.0, TiCDC OpenAPI becomes an General Availability (GA) feature. You can query and operate TiCDC clusters using OpenAPI in the production environment.

14.3.1.2.8 Deployment and maintenance

- **Continuous Profiling (experimental feature)**

TiDB Dashboard supports the Continuous Profiling feature, which stores instance performance analysis results automatically in real time when TiDB clusters are running. You can check the performance analysis result in a flame graph, which is more observable and shortens troubleshooting time.

This feature is disabled by default and needs to be enabled on the **Continuous Profile** page of TiDB Dashboard.

This feature is only available for clusters upgraded or installed using TiUP v1.7.0 or above.

[User document](#)

14.3.1.3 Telemetry

TiDB adds the information to the telemetry report about whether or not the TEMPORARY TABLE feature is used. This does not include table names or table data.

To learn more about telemetry and how to disable this behavior, refer to [Telemetry](#).

14.3.1.4 Removed feature

Starting from TiCDC v5.3.0, the cyclic replication feature between TiDB clusters (an experimental feature in v5.0.0) has been removed. If you have already used this feature to replicate data before upgrading TiCDC, the related data is not affected after the upgrade.

14.3.1.5 Improvements

- TiDB

- Show the affected SQL statements in the debug log when the coprocessor encounters a lock, which is helpful in diagnosing problems [#27718](#)
- Support showing the size of the backup and restore data when backing up and restoring data in the SQL logical layer [#27247](#)
- Improve the default collection logic of ANALYZE when `tidb_analyze_version` is 2, which accelerates collection and reduces resource overhead
- Introduce the `ANALYZE TABLE table_name COLUMNS col_1, col_2, ... , ↪ col_n` syntax. The syntax allows collecting statistics only on a portion of the columns in wide tables, which improves the speed of statistics collection

- TiKV

- Enhance disk space protection to improve storage stability

To solve the issue that TiKV might panic in case of a disk fully-written error, TiKV introduces a two-level threshold defense mechanism to protect the disk remaining space from being exhausted by excess traffic. Additionally, the mechanism provides the ability to reclaim space when the threshold is triggered. When the remaining space threshold is triggered, some write operations will fail and TiKV will return a disk full error as well as a list of disk full nodes. In this case, to recover the space and restore the service, you can execute `Drop/Truncate ↪ Table` or scale out the nodes.

- Simplify the algorithm of L0 flow control [#10879](#)
- Improve the error log report in the raft client module [#10944](#)
- Improve logging threads to avoid them becoming a performance bottleneck [#10841](#)
- Add more statistics types of write queries [#10507](#)

- PD

- Add more types of write queries to QPS dimensions in the hotspot scheduler [#3869](#)
- Support dynamically adjusting the retry limit of the balance Region scheduler to improve the performance of the scheduler [#3744](#)
- Update TiDB Dashboard to v2021.10.08.1 [#4070](#)
- Support that the evict leader scheduler can schedule Regions with unhealthy peers [#4093](#)
- Speed up the exit process of schedulers [#4146](#)
- TiFlash
 - Improve the execution efficiency of the TableScan operator greatly
 - Improve the execution efficiency of the Exchange operator
 - Reduce write amplification and memory usage during GC of the storage engine (experimental feature)
 - Improve the stability and availability of TiFlash when TiFlash restarts, which reduces possible query failures following the restart
 - Support pushing down multiple new String and Time functions to the MPP engine
 - * String functions: LIKE pattern, FORMAT(), LOWER(), LTRIM(), RTRIM(), SUBSTRING_INDEX(), TRIM(), UCASE(), UPPER()
 - * Mathematical functions: ROUND (decimal, int)
 - * Date and time functions: HOUR(), MICROSECOND(), MINUTE(), SECOND(), SYSDATE()
 - * Type conversion function: CAST(time, real)
 - * Aggregation functions: GROUP_CONCAT(), SUM(enum)
 - Support 512-bit SIMD
 - Enhance the cleanup algorithm for outdated data to reduce disk usage and read files more efficiently
 - Fix the issue that dashboard does not display memory or CPU information in some non-Linux systems
 - Unify the naming style of TiFlash log files (keep the naming style consistent with that of TiKV) and support dynamic modification of logger.count and logger.size
 - Improve the data validation capability of column-based files (checksums, experimental feature)
- Tools
 - TiCDC
 - * Reduce the default value of the Kafka sink configuration item MaxMessageBytes ↔ from 64 MB to 1 MB to fix the issue that large messages are rejected by the Kafka Broker [#3104](#)

- * Reduce memory usage in the replication pipeline [#2553](#) [#3037](#) [#2726](#)
- * Optimize monitoring items and alert rules to improve observability of synchronous links, memory GC, and stock data scanning processes [#2735](#) [#1606](#) [#3000](#) [#2985](#) [#2156](#)
- * When the sync task status is normal, no more historical error messages are displayed to avoid misleading users [#2242](#)

14.3.1.6 Bug Fixes

- TiDB
 - Fix an error that occurs during execution caused by the wrong execution plan. The wrong execution plan is caused by the shallow copy of schema columns when pushing down the aggregation operators on partitioned tables [#27797](#) [#26554](#)
 - Fix the issue that plan-cache cannot detect changes of unsigned flags [#28254](#)
 - Fix the wrong partition pruning when the partition function is out of range [#28233](#)
 - Fix the issue that planner might cache invalid plans for `join` in some cases [#28087](#)
 - Fix wrong index hash join when hash column type is enum [#27893](#)
 - Fix a batch client bug that recycling idle connection might block sending requests in some rare cases [#27688](#)
 - Fix the TiDB Lightning panic issue when it fails to perform checksum on a target cluster [#27686](#)
 - Fix wrong results of the `date_add` and `date_sub` functions in some cases [#27232](#)
 - Fix wrong results of the `hour` function in vectorized expression [#28643](#)
 - Fix the authenticating issue when connecting to MySQL 5.1 or an older client version [#27855](#)
 - Fix the issue that auto analyze might be triggered out of the specified time when a new index is added [#28698](#)
 - Fix a bug that setting any session variable invalidates `tidb_snapshot` [#28683](#)
 - Fix a bug that BR is not working for clusters with many missing-peer Regions [#27534](#)
 - Fix the unexpected error like `tidb_cast to Int32 is not supported` when the unsupported `cast` is pushed down to TiFlash [#23907](#)
 - Fix the issue that DECIMAL overflow is missing in the `%s` value is out of ↪ `range in '%s'` error message [#27964](#)
 - Fix a bug that the availability detection of MPP node does not work in some corner cases [#3118](#)
 - Fix the DATA RACE issue when assigning MPP task ID [#27952](#)
 - Fix the INDEX OUT OF RANGE error for a MPP query after deleting an empty dual table [#28250](#)
 - Fix the issue of false positive error log `invalid cop task execution summaries ↪ length` for MPP queries [#1791](#)
 - Fix the issue of error log `cannot found column in Schema column` for MPP queries [#28149](#)

- Fix the issue that TiDB might panic when TiFlash is shutting down [#28096](#)
 - Remove the support for insecure 3DES (Triple Data Encryption Algorithm) based TLS cipher suites [#27859](#)
 - Fix the issue that Lightning connects to offline TiKV nodes during pre-check and causes import failures [#27826](#)
 - Fix the issue that pre-check cost too much time when importing many files to tables [#27605](#)
 - Fix the issue that rewriting expressions makes `between` infer wrong collation [#27146](#)
 - Fix the issue that `group_concat` function did not consider the collation [#27429](#)
 - Fix the result wrong that occurs when the argument of the `extract` function is a negative duration [#27236](#)
 - Fix the issue that creating partition fails if `NO_UNSIGNED_SUBTRACTION` is set [#26765](#)
 - Avoid expressions with side effects in column pruning and aggregation pushdown [#27106](#)
 - Remove useless gRPC logs [#24190](#)
 - Limit the valid decimal length to fix precision-related issues [#3091](#)
 - Fix the issue of a wrong way to check for overflow in `plus` expression [#26977](#)
 - Fix the issue of `data too long` error when dumping statistics from the table with `new collation` data [#27024](#)
 - Fix the issue that the retried transactions' statements are not included in `TIDB_TRX` [#28670](#)
 - Fix the wrong default value of the `plugin_dir` configuration [#28084](#)
- TiKV
 - Fix the issue of unavailable TiKV caused by Raftstore deadlock when migrating Regions. The workaround is to disable the scheduling and restart the unavailable TiKV [#10909](#)
 - Fix the issue that CDC adds scan retries frequently due to the Congest error [#11082](#)
 - Fix the issue that the Raft connection is broken when the channel is full [#11047](#)
 - Fix the issue that batch messages are too large in Raft client implementation [#9714](#)
 - Fix the issue that some coroutines leak in `resolved_ts` [#10965](#)
 - Fix a panic issue that occurs to the coprocessor when the size of response exceeds 4 GiB [#9012](#)
 - Fix the issue that snapshot Garbage Collection (GC) misses GC snapshot files when snapshot files cannot be garbage collected [#10813](#)
 - Fix a panic issue caused by timeout when processing Coprocessor requests [#10852](#)
 - Fix a memory leak caused by monitoring data of statistics threads [#11195](#)
 - Fix a panic issue caused by getting the cgroup information from some platforms [#10980](#)

- Fix the issue of poor scan performance because MVCC Deletion versions are not dropped by compaction filter GC [#11248](#)
- PD
 - Fix the issue that PD incorrectly delete the peers with data and in pending status because the number of peers exceeds the number of configured peers [#4045](#)
 - Fix the issue that PD does not fix down peers in time [#4077](#)
 - Fix the issue that the scatter range scheduler cannot schedule empty Regions [#4118](#)
 - Fix the issue that the key manager cost too much CPU [#4071](#)
 - Fix the data race issue that might occur when setting configurations of hot Region scheduler [#4159](#)
 - Fix slow leader election caused by stucked Region syncer [#3936](#)
- TiFlash
 - Fix the issue of inaccurate TiFlash Store Size statistics
 - Fix the issue that TiFlash fails to start up on some platforms due to the absence of library `ns1`
 - Block the infinite wait of `wait index` when writing pressure is heavy (a default timeout of 5 minutes is added), which prevents TiFlash from waiting too long for data replication to provide services
 - Fix the slow and no result issues of the log search when the log volume is large
 - Fix the issue that only the most recent logs can be searched when searching old historical logs
 - Fix the possible wrong result when a new collation is enabled
 - Fix the possible parsing errors when an SQL statement contains extremely long nested expressions
 - Fix the possible `Block schema mismatch` error of the Exchange operator
 - Fix the possible `Can't compare` error when comparing Decimal types
 - Fix the `3rd arguments of function substringUTF8 must be constants` error of the `left/substring` function
- Tools
 - TiCDC
 - * Fix the issue that TiCDC replication task might terminate when the upstream TiDB instance unexpectedly exits [#3061](#)
 - * Fix the issue that TiCDC process might panic when TiKV sends duplicate requests to the same Region [#2386](#)
 - * Fix unnecessary CPU consumption when verifying downstream TiDB/MySQL availability [#3073](#)
 - * Fix the issue that the volume of Kafka messages generated by TiCDC is not constrained by `max-message-size` [#2962](#)

- * Fix the issue that TiCDC sync task might pause when an error occurs during writing a Kafka message [#2978](#)
 - * Fix the issue that some partitioned tables without valid indexes might be ignored when `force-replicate` is enabled [#2834](#)
 - * Fix the issue that scanning stock data might fail due to TiKV performing GC when scanning stock data takes too long [#2470](#)
 - * Fix a possible panic issue when encoding some types of columns into Open Protocol format [#2758](#)
 - * Fix a possible panic issue when encoding some types of columns into Avro format [#2648](#)
- TiDB Binlog
 - * Fix the issue that when most tables are filtered out, checkpoint can not be updated under some special load [#1075](#)

14.4 v5.2

14.4.1 TiDB 5.2.3 Release Note

Release date: December 3, 2021

TiDB version: 5.2.3

14.4.1.1 Bug fix

- TiKV
 - Fix the issue that the `GcKeys` task does not work when it is called by multiple keys. Caused by this issue, compaction filer GC might not drop the MVCC deletion information. [#11217](#)

14.4.2 TiDB 5.2.2 Release Notes

Release Date: October 29, 2021

TiDB version: 5.2.2

14.4.2.1 Improvements

- TiDB
 - Show the affected SQL statements in the debug log when the coprocessor encounters a lock, which is helpful in diagnosing problems [#27718](#)
 - Support showing the size of the backup and restore data when backing up and restoring data in the SQL logical layer [#27247](#)

- TiKV
 - Simplify the algorithm of L0 flow control [#10879](#)
 - Improve the error log report in the raft client module [#10983](#)
 - Improve logging threads to avoid them becoming a performance bottleneck [#10841](#)
 - Add more statistics types of write queries [#10507](#)
- PD
 - Add more types of write queries to QPS dimensions in the hotspot scheduler [#3869](#)
 - Support dynamically adjusting the retry limit of the balance region scheduler to improve the performance of the scheduler [#3744](#)
 - Update TiDB Dashboard to v2021.10.08.1 [#4070](#)
 - Support that the evict leader scheduler can schedule regions with unhealthy peers [#4093](#)
 - Speed up the exit process of schedulers [#4146](#)
- Tools
 - TiCDC
 - * Reduce the default value of the Kafka sink configuration item `MaxMessageBytes` ↗ from 64 MB to 1 MB to fix the issue that large messages are rejected by the Kafka Broker [#3104](#)
 - * Reduce memory usage in the replication pipeline [#2553](#) [#3037](#) [#2726](#)
 - * Optimize monitoring items and alert rules to improve observability of synchronous links, memory GC, and stock data scanning processes [#2735](#) [#1606](#) [#3000](#) [#2985](#) [#2156](#)
 - * When the sync task status is normal, no more historical error messages are displayed to avoid misleading users [#2242](#)

14.4.2.2 Bug Fixes

- TiDB
 - Fix the issue that plan-cache cannot detect changes of unsigned flags [#28254](#)
 - Fix the wrong partition pruning when the partition function is out of range [#28233](#)
 - Fix the issue that planner might cache invalid plans for `join` in some cases [#28087](#)
 - Fix wrong index hash join when hash column type is enum [#27893](#)
 - Fix a batch client bug that recycling idle connection might block sending requests in some rare cases [#27688](#)
 - Fix the TiDB Lightning panic issue when it fails to perform checksum on a target cluster [#27686](#)

- Fix wrong results of the `date_add` and `date_sub` functions in some cases [#27232](#)
- Fix wrong results of the `hour` function in vectorized expression [#28643](#)
- Fix the authenticating issue when connecting to MySQL 5.1 or an older client version [#27855](#)
- Fix the issue that auto analyze might be triggered out of the specified time when a new index is added [#28698](#)
- Fix a bug that setting any session variable invalidates `tidb_snapshot` [#28683](#)
- Fix a bug that BR is not working for clusters with many missing-peer regions [#27534](#)
- Fix the unexpected error like `tidb_cast` to `Int32` is not supported when the unsupported `cast` is pushed down to TiFlash [#23907](#)
- Fix the issue that `DECIMAL overflow` is missing in the `%s` value is out of ↳ `range` in '`%s`' error message [#27964](#)
- Fix a bug that the availability detection of MPP node does not work in some corner cases [#3118](#)
- Fix the DATA RACE issue when assigning MPP task ID [#27952](#)
- Fix the INDEX OUT OF RANGE error for a MPP query after deleting an empty dual table. [#28250](#)
- Fix the issue of false positive error log `invalid cop task execution summaries` ↳ `length` for MPP queries [#1791](#)
- Fix the issue of error log `cannot found column` in Schema column for MPP queries [#28149](#)
- Fix the issue that TiDB might panic when TiFlash is shutting down [#28096](#)
- Remove the support for insecure 3DES (Triple Data Encryption Algorithm) based TLS cipher suites [#27859](#)
- Fix the issue that Lightning connects to offline TiKV nodes during pre-check and causes import failures [#27826](#)
- Fix the issue that pre-check cost too much time when importing many files to tables [#27605](#)
- Fix the issue that rewriting expressions makes `between` infer wrong collation [#27146](#)
- Fix the issue that `group_concat` function did not consider the collation [#27429](#)
- Fix the result wrong that occurs when the argument of the `extract` function is a negative duration [#27236](#)
- Fix the issue that creating partition fails if `NO_UNSIGNED_SUBTRACTION` is set [#26765](#)
- Avoid expressions with side effects in column pruning and aggregation pushdown [#27106](#)
- Remove useless gRPC logs [#24190](#)
- Limit the valid decimal length to fix precision-related issues [#3091](#)
- Fix the issue of a wrong way to check for overflow in `plus` expression [#26977](#)
- Fix the issue of `data too long` error when dumping statistics from the table with `new collation` data [#27024](#)
- Fix the issue that the retried transactions' statements are not included in `TIDB_TRX` [#28670](#)

- TiKV
 - Fix the issue that CDC adds scan retries frequently due to the Congest error [#11082](#)
 - Fix the issue that the raft connection is broken when the channel is full [#11047](#)
 - Fix the issue that batch messages are too large in Raft client implementation [#9714](#)
 - Fix the issue that some coroutines leak in `resolved_ts` [#10965](#)
 - Fix a panic issue that occurs to the coprocessor when the size of response exceeds 4 GiB [#9012](#)
 - Fix the issue that snapshot Garbage Collection (GC) misses GC snapshot files when snapshot files cannot be garbage collected [#10813](#)
 - Fix a panic issue caused by timeout when processing Coprocessor requests [#10852](#)

- PD
 - Fix the issue that PD incorrectly delete the peers with data and in pending status because the number of peers exceeds the number of configured peers [#4045](#)
 - Fix the issue that PD does not fix down peers in time [#4077](#)
 - Fix the issue that the scatter range scheduler cannot schedule empty regions [#4118](#)
 - Fix the issue that the key manager cost too much CPU [#4071](#)
 - Fix the data race issue that might occur when setting configurations of hot region scheduler [#4159](#)
 - Fix slow leader election caused by stuck region syncer [#3936](#)

- TiFlash
 - Fix the issue that TiFlash fails to start up on some platforms due to the absence of library `ns1`

- Tools
 - TiCDC
 - * Fix the issue that TiCDC replication task might terminate when the upstream TiDB instance unexpectedly exits [#3061](#)
 - * Fix the issue that TiCDC process might panic when TiKV sends duplicate requests to the same Region [#2386](#)
 - * Fix unnecessary CPU consumption when verifying downstream TiDB/MySQL availability [#3073](#)
 - * Fix the issue that the volume of Kafka messages generated by TiCDC is not constrained by `max-message-size` [#2962](#)
 - * Fix the issue that TiCDC sync task might pause when an error occurs during writing a Kafka message [#2978](#)
 - * Fix the issue that some partitioned tables without valid indexes might be ignored when `force-replicate` is enabled [#2834](#)

- * Fix the issue that scanning stock data might fail due to TiKV performing GC when scanning stock data takes too long [#2470](#)
- * Fix a possible panic issue when encoding some types of columns into Open Protocol format [#2758](#)
- * Fix a possible panic issue when encoding some types of columns into Avro format [#2648](#)
- TiDB Binlog
 - * Fix the issue that when most tables are filtered out, checkpoint can not be updated under some special load [#1075](#)

14.4.3 TiDB 5.2.1 Release Notes

Release date: September 9, 2021

TiDB version: 5.2.1

14.4.3.1 Bug fixes

- TiDB
 - Fix an error that occurs during execution caused by the wrong execution plan. The wrong execution plan is caused by the shallow copy of schema columns when pushing down the aggregation operators on partitioned tables. [#27797](#) [#26554](#)
- TiKV
 - Fix the issue of unavailable TiKV caused by Raftstore deadlock when migrating Regions. The workaround is to disable the scheduling and restart the unavailable TiKV. [#10909](#)

14.4.4 TiDB 5.2 Release Notes

Release date: August 27, 2021

TiDB version: 5.2.0

Warning:

Some known issues are found in this version, and these issues are fixed in new versions. It is recommended that you use the latest 5.2.x version.

In v5.2, the key new features and improvements are as follows:

- Support using several functions in expression indexes to greatly improve query performance
- Improve the accuracy of optimizer cardinality estimation to help to select optimal execution plans
- Announce the general availability (GA) for the Lock View feature to observe transaction locking events and troubleshoot deadlock problems
- Add the TiFlash I/O traffic limit feature to improve the stability of read and write for TiFlash
- TiKV introduces a new flow control mechanism to replace the previous RocksDB write stall mechanism to improve the stability of TiKV flow control
- Simplify the operation and maintenance of Data Migration (DM) to reduce the management cost.
- TiCDC supports HTTP protocol OpenAPI to manage TiCDC tasks. It provides a more user-friendly operation method for both Kubernetes and on-premises environments. (Experimental feature)

14.4.4.1 Compatibility changes

Note:

When upgrading from an earlier TiDB version to v5.2, if you want to know the compatibility change notes of all intermediate versions, you can check the [Release Note](#) for the corresponding version.

14.4.4.1.1 System variables

Variable name	Change type	Description
<code>default_authen</code> → <code>new_authentication_plugins</code>	Newly added plugin	Sets the authentication method that the server advertises. The default value is <code>mysql_native_password</code> → .

Variable	Change type	Description
<code>tidb_enable_auto_indexed</code>	Newly added	<code>Determinated</code> whether to include the <code>AUTO_INCREMENT</code> columns when creating a generated column or an expression index. The default value is <code>OFF</code> .
<code>tidb_opt_enable_index_selection</code>	Newly added	<code>Controlment</code> whether the optimizer estimates the number of rows based on column order correlation. The default value is <code>ON</code> .
<code>tidb_opt_limit_pushdown_threshold</code>	Newly added	<code>Sets threshold</code> threshold that determines whether to push the Limit or TopN operator down to TiKV. The default value is 100.

Variable name	Change type	Description
<code>tidb_stmt_summary</code>	Modified → <code>stmt_Setsum</code>	maximum number of statements that the statement summary tables store in memory. The default value is changed from 200 to 3000.
<code>tidb_enable_streamingd</code>	Dropped → <code>enable-streaming</code>	The system variable enable- → streaming → is deprecated and it is not recommended to use it any more.

14.4.4.1.2 Configuration file parameters

Configuration file	Configuration item	Change type	Description
TiDB configuration file	<code>pessimistic</code> ↪ <code>-txn.</code> ↪ <code>deadlock</code> ↪ <code>-</code> ↪ <code>history</code> ↪ <code>-</code> ↪ <code>collect</code> ↪ <code>-</code> ↪ <code>retryable</code> ↪	Newly added	Controls whether the <code>INFORMATION_SCHEMA</code> ↪ . ↪ <code>DEADLOCKS</code> ↪ table collects retryable deadlock error messages or not.

Configuration file	Configuration item	Change type	Description
TiDB configuration file	<code>security.</code> ↳ <code>auto-</code> ↳ <code>tls</code>	Newly added	Determines whether to automatically generate the TLS certificates on startup. The default value is <code>false</code> .
TiDB configuration file	<code>stmt-</code> ↳ <code>summary</code> ↳ <code>.max-</code> ↳ <code>stmt-</code> ↳ <code>count</code>	Modified	Indicates the maximum number of SQL categories allowed to be saved in the statement summary tables. The default value is changed from 200 to 3000.
TiDB configuration file	<code>experimental</code> ↳ <code>.allow-</code> ↳ <code>expression</code> ↳ <code>-index</code>	Deprecated	The <code>allow-</code> ↳ <code>expression</code> ↳ <code>-index</code> configuration in the TiDB configuration file is deprecated.

Configuration file	Configuration item	Change type	Description
TiKV configuration file	raftstore. ↳ cmd- ↳ batch	Newly added	<p>Controls whether to enable batch processing of the requests. When it is enabled, the write performance is significantly improved. The default value is true.</p>
TiKV configuration file	raftstore. ↳ inspect ↳ - ↳ interval ↳	Newly added	<p>At a certain interval, TiKV inspects the latency of the Raftstore component. This configuration item specifies the interval of the inspection. If the latency exceeds this value, this inspection is marked as timeout. The default value is 500ms.</p>

Configuration file	Configuration item	Change type	Description
TiKV configuration file	<code>raftstore.</code> ↳ <code>max-</code> ↳ <code>peer-</code> ↳ <code>down-</code> ↳ <code>duration</code> ↳	Modified	Indicates the longest inactive duration allowed for a peer. A peer with timeout is marked as down, and PD tries to delete it later. The default value is changed from 5m to 10m.
TiKV configuration file	<code>server.</code> ↳ <code>raft-</code> ↳ <code>client-</code> ↳ <code>queue-</code> ↳ <code>size</code>	Newly added	Specifies the queue size of the Raft messages in TiKV. The default value is 8192.
TiKV configuration file	<code>storage.</code> ↳ <code>flow-</code> ↳ <code>control</code> ↳ <code>.enable</code>	Newly added	Determines whether to enable the flow control mechanism. The default value is true.

file	item	Change type	Description
TiKV configuration file	storage. ↳ flow- ↳ control ↳ . ↳ memtables ↳ - ↳ threshold ↳	Newly added	When the number of kvDB memtables reaches this threshold, the flow control mechanism starts to work. The default value is 5.
TiKV configuration file	storage. ↳ flow- ↳ control ↳ .10- ↳ files- ↳ threshold ↳	Newly added	When the number of kvDB L0 files reaches this threshold, the flow control mechanism starts to work. The default value is 9.

Configuration file	Configuration item	Change type	Description
TiKV configuration file	<code>storage.</code> \hookrightarrow <code>flow-</code> \hookrightarrow <code>control</code> \hookrightarrow <code>.soft-</code> \hookrightarrow <code>pending</code> \hookrightarrow <code>-</code> \hookrightarrow <code>compaction</code> \hookrightarrow <code>-bytes-</code> \hookrightarrow <code>limit</code>	Newly added	When the pending compaction bytes in KvDB reach this threshold, the flow control mechanism starts to reject some write requests and reports the <code>ServerIsBusy</code> \hookrightarrow error. The default value is “192GB”.
TiKV configuration file	<code>storage.</code> \hookrightarrow <code>flow-</code> \hookrightarrow <code>control</code> \hookrightarrow <code>.hard-</code> \hookrightarrow <code>pending</code> \hookrightarrow <code>-</code> \hookrightarrow <code>compaction</code> \hookrightarrow <code>-bytes-</code> \hookrightarrow <code>limit</code>	Newly added	When the pending compaction bytes in KvDB reach this threshold, the flow control mechanism rejects all write requests and reports the <code>ServerIsBusy</code> \hookrightarrow error. The default value is “1024GB”.

14.4.4.1.3 Others

- Before the upgrade, check whether the value of the `tidb_evolve_plan_baselines` system variable is `ON`. If the value is `ON`, set it to `OFF`; otherwise, the upgrade will fail.
- For TiDB clusters upgraded from v4.0 to v5.2, the default value of `tidb_multi_statement_mode` changes from `WARN` to `OFF`.
- Before the upgrade, check the value of the TiDB configuration `feedback-probability`. If the value is not 0, the “panic in the recoverable goroutine” error will occur after the upgrade, but this error does not affect the upgrade.
- TiDB is now compatible with MySQL 5.7’s noop variable `innodb_default_row_format`. Setting this variable has no effect. [#23541](#)
- Starting from TiDB 5.2, to improve system security, it is recommended (but not mandatory) to encrypt the transport layer for connections from clients. TiDB provides the Auto TLS feature to automatically configure and enable encryption in TiDB. To use the Auto TLS feature, before the TiDB upgrade, set `security.auto-tls` in the TiDB configuration file to `true`.
- Support the `caching_sha2_password` authentication method to make migration from MySQL 8.0 easier and to improve security.

14.4.4.2 New features

14.4.4.2.1 SQL

- **Support using several functions in expression indexes**

The expression index is a type of special index that can be created on an expression. After an expression index is created, TiDB supports expression-based queries, which greatly improves query performance.

[User document](#), [#25150](#)

- **Support the `translate` function in Oracle**

The `translate` function replaces all occurrences of characters by other characters in a string. In TiDB, this function does not treat empty strings as `NULL` as Oracle does.

[User document](#)

- **Support spilling HashAgg**

Support spilling HashAgg into disk. When a SQL statement that includes an HashAgg operator causes out of memory (OOM), you can try to set the concurrency of this operator to 1 to trigger disk spill, which alleviates memory stress.

[User document](#), [#25882](#)

- **Improve the accuracy of optimizer cardinality estimation**

– Improve the accuracy of TiDB’s estimation of TopN/Limit. For example, for pagination queries on a large table that contain the `order by col limit x` condition, TiDB can more easily select the right index and reduce query response time.

- Improve the accuracy of out-of-range estimation. For example, even if the statistics for a day have not been updated, TiDB can accurately select the corresponding index for a query that contains `where date=Now()`.
- Introduce the `tidb_opt_limit_push_down_threshold` variable to control the optimizer's behavior of pushing down Limit/TopN, which resolves the issue that Limit/TopN cannot be pushed down in some situations due to wrong estimation.

[User document](#), #26085

- **Improve index selection of the optimizer**

Add pruning rules for index selection. Before using the statistics for comparison, TiDB uses these rules to narrow down the scope of possible indexes to be selected, which reduces the possibility of selecting non-optimal indexes.

[User document](#)

14.4.4.2.2 Transaction

- **General availability (GA) for Lock View**

The Lock View feature provides more information about lock conflicts and lock waits of pessimistic locks, which helps DBAs to observe transaction locking events and troubleshoot deadlock problems.

In v5.2, the following enhancements are made to Lock View:

- In addition to the SQL digest column in the Lock View-related tables, add a column to these tables that shows the corresponding normalized SQL text. You do not have to manually query the statement corresponding to a SQL digest.
- Add the `TIDB_DECODE_SQL_DIGESTS` function to query the normalized SQL statements (a form without formats and arguments) corresponding to a set of SQL digests in the cluster. This simplifies the operation of querying the statements that have been historically executed by a transaction.
- Add a column in the `DATA_LOCK_WAIT`s and `DEADLOCKS` system tables to show the table name, row ID, index value, and other key information interpreted from a key. This simplifies the operations such as locating the table to which a key belongs and interpreting the key information.
- Support collecting the information of retryable deadlock errors in the `DEADLOCKS` table, which makes it easier to troubleshoot issues caused by such errors. The error collection is disabled by default and can be enabled using the `pessimistic ↳ -txn.deadlock-history-collect-retryable` configuration.
- Support distinguishing query-executing transactions from idle transactions on the `TIDB_TRX` system table. The `Normal` state is now divided into `Running` and `Idle` states.

User documents:

- View the pessimistic lock-waiting events that are occurring on all TiKV nodes in the cluster: [`DATA_LOCK_WAIT`] (#data_lock_waits)
- View the deadlock errors recently occurred on a TiDB node: [`DEADLOCKS`] (#deadlocks)
- View the executing transaction on a TiDB node: [`TIDB_TRX`] (#tidb_trx)

- Optimize the user scenarios of adding indexes on tables with the AUTO_RANDOM or SHARD_ROW_ID_BITS attribute.

14.4.4.2.3 Stability

- **Add TiFlash I/O traffic limit**

This new feature is suitable for cloud storage with disk bandwidth of a small and specific size. It is disabled by default.

TiFlash I/O Rate Limiter provides a new mechanism to avoid excessive race for I/O resources between read and write tasks. It balances the responses to read and write tasks, and limits the rate automatically according to read/write workload.

[User document](#)

- **Improve stability of TiKV flow control**

TiKV introduces a new flow control mechanism to replace the previous RocksDB write stall mechanism. Compared with the write stall mechanism, this new mechanism reduces the impact on the stability of foreground write.

In specific, when the stress of RocksDB compaction accumulates, flow control is performed at the TiKV scheduler layer instead of the RocksDB layer, to avoid the following issues:

- Raftstore is stuck, which is caused by RocksDB write stall.
- Raft election times out, and the node leader is transferred as a result.

This new mechanism improves the flow control algorithm to mitigate QPS decrease when the write traffic is high.

[User document, #10137](#)

- **Detect and recover automatically from impact caused by a single slow TiKV node in a cluster**

TiKV introduces the slow node detection mechanism. This mechanism calculates a score by inspecting the rate of TiKV Raftstore, and then reports the score to PD through store heartbeats. Meanwhile, it adds the evict-slow-store-scheduler scheduler on PD to automatically evict the leader on a single slow TiKV node. In this way, the impact on the whole cluster is mitigated. At the same time, more alert items about slow nodes are introduced to help you quickly pinpoint and solve problems.

[User document, #10539](#)

14.4.4.2.4 Data Migration

- **Simplify operations of Data Migration (DM)**

DM v2.0.6 can automatically identify the change event (failover or plan change) of the data source using VIP, and can automatically connect to a new data source instance, to reduce data replication latency and simplify operation procedures.

- TiDB Lightning supports customized line terminators in the CSV data, and is compatible with the MySQL LOAD DATA CSV data formats. You can then use TiDB Lightning directly in your data flow architecture.

[#1297](#)

14.4.4.2.5 TiDB data share subscription

TiCDC supports using the HTTP protocol (OpenAPI) to manage TiCDC tasks, which is a more user-friendly operation method for both Kubernetes and on-premises environments. (Experimental feature)

[#2411](#)

14.4.4.2.6 Deployment and operations

Support running the `tiup playground` command on Mac computers with Apple M1 chips.

14.4.4.3 Feature Enhancements

- Tools

- TiCDC

- * Add the binary MQ format designed for TiDB. It is more compact than the open protocols based on JSON [#1621](#)
 - * Remove support for file sorter [#2114](#)
 - * Support log rotation configurations [#2182](#)

- TiDB Lightning

- * Support customized line terminators (except \r and \n) [#1297](#)
 - * Support expression index and the index that depends on virtual generated columns [#1407](#)

- Dumpling

- * Support backing up MySQL compatible databases but does not support START TRANSACTION ... WITH CONSISTENT SNAPSHOT or SHOW CREATE → TABLE [#311](#)

14.4.4.4 Improvements

- TiDB
 - Support pushing down the built-in function `json_unquote()` to TiKV [#24415](#)
 - Support removing the `union` branch from the dual table [#25614](#)
 - Optimize the aggregate operator's cost factor [#25241](#)
 - Allow the MPP outer join to choose the build table based on the table row count [#25142](#)
 - Support balancing the MPP query workload among different TiFlash nodes based on Regions [#24724](#)
 - Support invalidating stale Regions in the cache after the MPP query is executed [#24432](#)
 - Improve the MySQL compatibility of the built-in function `str_to_date` for the format specifiers `%b/%M/%r/%T` [#25767](#)
 - Fix the issue that inconsistent binding caches might be created in multiple TiDB after recreating different bindings for the same query [#26015](#)
 - Fix the issue that the existing bindings cannot be loaded into cache after upgrade [#23295](#)
 - Support ordering the result of `SHOW BINDINGS` by `(original_sql, update_time)` [#26139](#)
 - Improve the logic of query optimization when bindings exist, and reduce optimization times of a query [#26141](#)
 - Support completing the garbage collection automatically for the bindings in the “deleted” status [#26206](#)
 - Support showing whether a binding is used for query optimization in the result of `EXPLAIN VERBOSE` [#26930](#)
 - Add a new status variation `last_plan_binding_update_time` to view the timestamp corresponding to the binding cache in the current TiDB instance [#26340](#)
 - Support reporting an error when starting binding evolution or running `admin ↗ evolve bindings` to ban the baseline evolution (currently disabled in the on-premises TiDB version because it is an experimental feature) affecting other features [#26333](#)
- PD
 - Add more QPS dimensions for hot Region scheduling, and support adjusting the priority of the scheduling [#3869](#)
 - Support hot Region balance scheduling for the write hotspot of TiFlash [#3900](#)
- TiFlash
 - Add operators: `MOD` / `%`, `LIKE`
 - Add string functions: `ASCII()`, `COALESCE()`, `LENGTH()`, `POSITION()`, `TRIM()`
 - Add mathematical functions: `CONV()`, `CRC32()`, `DEGREES()`, `EXP()`, `LN()`, `LOG()`, `LOG10()`, `LOG2()`, `POW()`, `RADIANS()`, `ROUND(decimal)`, `SIN()`, `MOD()`

- Add date functions: `ADDDATE(string, real)`, `DATE_ADD(string, real)`, `DATE → ()`
- Add other functions: `INET_NTOA()`, `INET_ATON()`, `INET6_ATON`, `INET6_NTOA()`
- Support Shuffled Hash Join calculation and Shuffled Hash Aggregation calculation in the MPP mode when a new collation is enabled
- Optimize basic code to improve MPP performance
- Support casting the `STRING` type to the `DOUBLE` type
- Optimize the non-joined data in right outer join using multiple threads
- Support automatically invalidating stale Regions in MPP queries

- Tools

- TiCDC
 - * Add the concurrency limit to the incremental scan of kv client [#1899](#)
 - * TiCDC can always pull the old value internally [#2271](#)
 - * TiCDC can fail and exit fast when unrecoverable DML errors occur [#1928](#)
 - * `resolve lock` cannot be run immediately after a Region is initialized [#2235](#)
 - * Optimize workerpool to reduce the number of goroutines under high concurrency [#2201](#)
- Dumpling
 - * Support always splitting TiDB v3.x tables through `tidb_rowid` to save TiDB memory [#301](#)
 - * Reduce access of Dumpling to the `information_schema` to improve stability [#305](#)

14.4.4.5 Bug Fixes

- TiDB

- Fix the issue that an incorrect result is returned when using merge join on the `SET` type column [#25669](#)
- Fix the data corruption issue in the `IN` expression's arguments [#25591](#)
- Avoid the sessions of GC being affected by global variables [#24976](#)
- Fix the panic issue that occurs when using `limit` in the window function queries [#25344](#)
- Fix the wrong value returned when querying a partitioned table using `Limit #24636`
- Fix the issue that `IFNULL` does not correctly take effect on the `ENUM` or `SET` type column [#24944](#)
- Fix the wrong results caused by changing the `count` in the join subqueries to `first_row` [#24865](#)
- Fix the query hang issue that occurs when `ParallelApply` is used under the `TopN` operator [#24930](#)

- Fix the issue that more results than expected are returned when executing SQL statements using multi-column prefix indexes [#24356](#)
- Fix the issue that the `<=>` operator cannot correctly take effect [#24477](#)
- Fix the data race issue of the parallel `Apply` operator [#23280](#)
- Fix the issue that the `index out of range` error is reported when sorting the `IndexMerge` results of the `PartitionUnion` operator [#23919](#)
- Fix the issue that setting the `tidb_snapshot` variable to an unexpectedly large value might damage the transaction isolation [#25680](#)
- Fix the issue that the ODBC-styled constant (for example, `{d '2020-01-01'}`) cannot be used as the expression [#25531](#)
- Fix the issue that `SELECT DISTINCT` converted to `Batch Get` causes incorrect results [#25320](#)
- Fix the issue that backing off queries from TiFlash to TiKV cannot be triggered [#23665 #24421](#)
- Fix the `index-out-of-range` error that occurs when checking `only_full_group_by` ↪ [#23839](#)
- Fix the issue that the result of index join in correlated subqueries is wrong [#25799](#)

- TiKV

- Fix the wrong `tikv_raftstore_hibernated_peer_state` metric [#10330](#)
- Fix the wrong arguments type of the `json_unquote()` function in the coprocessor [#10176](#)
- Skip clearing callback during graceful shutdown to avoid breaking ACID in some cases [#10353 #10307](#)
- Fix a bug that the read index is shared for replica reads on a Leader [#10347](#)
- Fix the wrong function that casts `DOUBLE` to `DOUBLE` [#25200](#)

- PD

- Fix the issue that the expected scheduling cannot be generated due to scheduling conflicts among multiple schedulers [#3807 #3778](#)

- TiFlash

- Fix the issue that TiFlash keeps restarting because of the split failure
- Fix the potential issue that TiFlash cannot delete the delta data
- Fix a bug that TiFlash adds wrong padding for non-binary characters in the `CAST` function
- Fix the issue of incorrect results when handling aggregation queries with complex `GROUP BY` columns
- Fix the TiFlash panic issue that occurs under heavy write pressure
- Fix the panic that occurs when the right join key is not nullable and the left join key is nullable
- Fix the potential issue that the `read-index` requests take a long time
- Fix the panic issue that occurs when the read load is heavy

- Fix the panic issue that might occur when the `Date_Format` function is called with the `STRING` type argument and `NULL` values
- Tools
 - TiCDC
 - * Fix a bug that TiCDC owner exits abnormally when refreshing the checkpoint [#1902](#)
 - * Fix a bug that changefeed fails immediately after its successful creation [#2113](#)
 - * Fix a bug that changefeed fails due to the invalid format of rules filter [#1625](#)
 - * Fix the potential DDL loss issue when the TiCDC owner panics [#1260](#)
 - * Fix the CLI compatibility issue with 4.0.x clusters on the default sort-engine option [#2373](#)
 - * Fix a bug that changefeed might be reset unexpectedly when TiCDC gets the `ErrSchemaStorageTableMiss` error [#2422](#)
 - * Fix a bug that changefeed cannot be removed when TiCDC gets the `ErrGCTTLExceeded` error [#2391](#)
 - * Fix a bug that TiCDC fails to synchronize large tables to cdclog [#1259](#) [#2424](#)
 - * Fix a bug that multiple processors might write data to the same table when TiCDC is rescheduling the table [#2230](#)
 - Backup & Restore (BR)
 - * Fix a bug that BR skips restoring all system tables during the restore [#1197](#) [#1201](#)
 - * Fix a bug that BR misses DDL operations when restoring cdclog [#870](#)
 - TiDB Lightning
 - * Fix a bug that TiDB Lightning fails to parse the `DECIMAL` data type in Parquet file [#1272](#)
 - * Fix a bug that TiDB Lightning reports the “Error 9007: Write conflict” error when restoring table schemas [#1290](#)
 - * Fix a bug that TiDB Lightning fails to import data due to the overflow of int handle [#1291](#)
 - * Fix a bug that TiDB Lightning might get a checksum mismatching error due to data loss in the local backend mode [#1403](#)
 - * Fix the Lightning incompatibility issue with clustered index when TiDB Lightning is restoring table schemas [#1362](#)
 - Dumpling
 - * Fix a bug that the data export fails because the Dumpling GC safepoint is set too late [#290](#)
 - * Fix the Dumpling getting stuck issue when exporting table names from the upstream database in certain MySQL versions [#322](#)

14.5 v5.1

14.5.1 TiDB 5.1.3 Release Note

Release date: December 3, 2021

TiDB version: 5.1.3

14.5.1.1 Bug fix

- TiKV
 - Fix the issue that the `GcKeys` task does not work when it is called by multiple keys. Caused by this issue, compaction filer GC might not drop the MVCC deletion information. [#11217](#)

14.5.2 TiDB 5.1.2 Release Notes

Release Date: September 27, 2021

TiDB version: 5.1.2

14.5.2.1 Compatibility changes

- TiDB
 - The following bug fixes change execution results, which might cause upgrade incompatibilities:
 - * Fix the issue that `greatest(datetime)union null` returns empty string [#26532](#)
 - * Fix the issue that the `having` clause might not work correctly [#26496](#)
 - * Fix the wrong execution results that occur when the collations around the `between` expression are different [#27146](#)
 - * Fix the wrong execution results that occur when the column in the `group_concat` function has a non-bin collation [#27429](#)
 - * Fix an issue that using a `count(distinct)` expression on multiple columns returns wrong result when the new collation is enabled [#27091](#)
 - * Fix the result wrong that occurs when the argument of the `extract` function is a negative duration [#27236](#)
 - * Fix the issue that inserting an invalid date does not report an error when the `SQL_MODE` is ‘`STRICT_TRANS_TABLES`’ [#26762](#)
 - * Fix the issue that using an invalid default date does not report an error when the `SQL_MODE` is ‘`NO_ZERO_IN_DATE`’ [#26766](#)
- Tools

- TiCDC
 - * Set the compatible version from 5.1.0-alpha to 5.2.0-alpha [#2659](#)

14.5.2.2 Improvements

- TiDB
 - Trigger auto-analyze by histogram row count and increase the accuracy of this trigger action [#24237](#)
- TiKV
 - Support dynamically modifying TiCDC configurations [#10645](#)
 - Reduce the size of Resolved TS message to save network bandwidth [#2448](#)
 - Limit the counts of peer stats in the heartbeat message reported by a single store [#10621](#)
- PD
 - Allow empty Regions to be scheduled and use a separate tolerance configuration in scatter range scheduler [#4117](#)
 - Improve the performance of synchronizing Region information between PDs [#3933](#)
 - Support dynamically adjusting the retry limit of a store based on the generated operator [#3744](#)
- TiFlash
 - Support the DATE() function
 - Add Grafana panels for write throughput per instance
 - Optimize the performance of the leader-read process
 - Accelerate the process of canceling MPP tasks
- Tools
 - TiCDC
 - * Optimize memory management when the Unified Sorter is using memory to sort data [#2553](#)
 - * Optimize workerpool for fewer goroutines when concurrency is high [#2211](#)
 - * Reduce goroutine usage when a table's Region transfer away from a TiKV node [#2284](#)
 - * Add a global gRPC connection pool and share gRPC connections among KV clients [#2534](#)
 - * Prohibit operating TiCDC clusters across major and minor versions [#2599](#)

- Dumpling
 - * Support backing up MySQL-compatible databases that do not support `START TRANSACTION ... WITH CONSISTENT SNAPSHOT` and `SHOW CREATE TABLE` [#309](#)

14.5.2.3 Bug fixes

- TiDB
 - Fix the potential wrong results of index hash join when the hash column is the `ENUM` type [#27893](#)
 - Fix a batch client bug that recycle idle connection might block sending requests in some rare cases [#27678](#)
 - Fix the issue that the overflow check of the `FLOAT64` type is different with that of MySQL [#23897](#)
 - Fix the issue that TiDB returns an `unknow` error while it should return the `pd → is timeout` error [#26147](#)
 - Fix the wrong character set and collation for the `case when` expression [#26662](#)
 - Fix the potential `can not found column` in Schema column error for MPP queries [#28148](#)
 - Fix a bug that TiDB might panic when TiFlash is shutting down [#28096](#)
 - Fix the issue of wrong range caused by using `enum like 'x%'` [#27130](#)
 - Fix the Common Table Expression (CTE) dead lock issue when used with `IndexLookupJoin` [#27410](#)
 - Fix a bug that retryable deadlocks are incorrectly recorded into the `INFORMATION_SCHEMA .DEADLOCKS` table [#27400](#)
 - Fix the issue that the `TABLESAMPLE` query result from partitioned tables is not sorted as expected [#27349](#)
 - Remove the unused `/debug/sub-optimal-plan` HTTP API [#27265](#)
 - Fix a bug that the query might return wrong results when the hash partitioned table deals with unsigned data [#26569](#)
 - Fix a bug that creating partition fails if `NO_UNSIGNED_SUBTRACTION` is set [#26765](#)
 - Fix the issue that the `distinct` flag is missing when `Apply` is converted to `Join` [#26958](#)
 - Set a block duration for the newly recovered TiFlash node to avoid blocking queries during this time [#26897](#)
 - Fix a bug that might occur when the CTE is referenced more than once [#26212](#)
 - Fix a CTE bug when `MergeJoin` is used [#25474](#)
 - Fix a bug that the `SELECT FOR UPDATE` statement does not correctly lock the data when a normal table joins a partitioned table [#26251](#)
 - Fix the issue that the `SELECT FOR UPDATE` statement returns an error when a normal table joins a partitioned table [#26250](#)
 - Fix the issue that `PointGet` does not use the lite version of resolving lock [#26562](#)
- TiKV

- Fix a panic issue that occurs after TiKV is upgraded from v3.x to later versions [#10902](#)
- Fix the potential disk full issue caused by corrupted snapshot files [#10813](#)
- Make the slow log of TiKV coprocessor only consider the time spent on processing requests [#10841](#)
- Drop log instead of blocking threads when the slogger thread is overloaded and the queue is filled up [#10841](#)
- Fix a panic issue that occurs when processing Coprocessor requests times out [#10852](#)
- Fix the TiKV panic issue that occurs when upgrading from a pre-5.0 version with Titan enabled [#10842](#)
- Fix the issue that TiKV of a newer version cannot be rolled back to v5.0.x [#10842](#)
- Fix the issue that TiKV might delete files before ingesting data to RocksDB [#10438](#)
- Fix the parsing failure caused by the left pessimistic locks [#26404](#)

- PD

- Fix the issue that PD does not fix the down peers in time [#4077](#)
- Fix the issue that the replica count of the default placement rules stays constant after `replication.max-replicas` is updated [#3886](#)
- Fix a bug that PD might panic when scaling out TiKV [#3868](#)
- Fix a bug that the hot Region scheduler cannot work when the cluster has the evict leader scheduler [#3697](#)

- TiFlash

- Fix the issue of unexpected results when TiFlash fails to establish MPP connections
- Fix the potential issue of data inconsistency that occurs when TiFlash is deployed on multiple disks
- Fix a bug that MPP queries get wrong results when TiFlash server is under high load
- Fix a potential bug that MPP queries hang forever
- Fix the panic issue when operating store initialization and DDL simultaneously
- Fix a bug of incorrect results that occurs when queries contain filters like `CONSTANT ↗, <, <=, >, >=,` or `COLUMN`
- Fix the potential panic issue when `Snapshot` is applied simultaneously with multiple DDL operations
- Fix the issue that the store size in metrics is inaccurate under heavy writing
- Fix the potential issue that TiFlash cannot garbage-collect the delta data after running for a long time
- Fix the issue of wrong results when the new collation is enabled
- Fix the potential panic issue that occurs when resolving locks
- Fix a potential bug that metrics display wrong values

- Tools
 - Backup & Restore (BR)
 - * Fix the issue that the average speed is not accurate during data backup and restore [#1405](#)
 - Dumpling
 - * Fix the issue that Dumpling is pending when `show table status` returns incorrect results in some MySQL versions (8.0.3 and 8.0.23) [#322](#)
 - * Fix the CLI compatibility issue with 4.0.x clusters on the default `sort-<-> engine` option [#2373](#)
 - TiCDC
 - * Fix a bug that the JSON encoding might cause panic when processing a string type value that is `string` or `[]byte` [#2758](#)
 - * Reduce gRPC window size to avoid OOM [#2673](#)
 - * Fix a gRPC `keepalive` error under high memory pressure [#2202](#)
 - * Fix a bug that an unsigned `tinyint` causes TiCDC to panic [#2648](#)
 - * Fix an empty value issue in TiCDC Open Protocol. An empty value is no longer output when there is no change in one transaction. [#2612](#)
 - * Fix a bug in DDL handling during manual restarts [#2603](#)
 - * Fix the issue that `EtcdbWorker`'s snapshot isolation might be wrongly violated when managing the metadata [#2559](#)
 - * Fix a bug that multiple processors might write data to the same table when TiCDC is rescheduling the table [#2230](#)
 - * Fix a bug that changefeed might be reset unexpectedly when TiCDC gets the `ErrSchemaStorageTableMiss` error [#2422](#)
 - * Fix a bug that changefeed cannot be removed when TiCDC gets the `ErrGCTTLExceeded` error [#2391](#)
 - * Fix a bug that TiCDC fails to synchronize large tables to cdclog [#1259](#) [#2424](#)

14.5.3 TiDB 5.1.1 Release Notes

Release Date: July 30, 2021

TiDB version: 5.1.1

14.5.3.1 Compatibility changes

- TiDB
 - For TiDB clusters upgrade from v4.0 to v5.1, the default value of `tidb_multi_statement_mode` is OFF. It is recommended to use the multi-statement feature of your client library instead. See [the documentation on tidb_multi_statement_mode](#) for details. [#25751](#)

- Change the default value of the `tidb_stmt_summary_max_stmt_count` variable from 200 to 3000 [#25874](#)
- Require the SUPER privilege to access the `table_storage_stats` table [#26352](#)
- Require the SELECT privilege on `mysql.user` to access the `information_schema`
 \hookrightarrow `.user_privileges` table to show other user's privileges [#26311](#)
- Require the CONFIG privilege to access the `information_schema.cluster_hardware`
 \hookrightarrow table [#26297](#)
- Require the PROCESS privilege to access the `information_schema.cluster_info`
 table [#26297](#)
- Require the PROCESS privilege to access the `information_schema.cluster_load`
 table [#26297](#)
- Require the PROCESS privilege to access the `information_schema.cluster_systeminfo`
 \hookrightarrow table [#26297](#)
- Require the PROCESS privilege to access the `information_schema.cluster_log`
 table [#26297](#)
- Require the CONFIG privilege to access the `information_schema.cluster_config`
 \hookrightarrow table [#26150](#)

14.5.3.2 Feature enhancements

- TiDB Dashboard
 - Support OIDC SSO. By setting the OIDC-compatible SSO services (such as Okta and Auth0), users can log into TiDB Dashboard without entering the SQL password. [#3883](#)
- TiFlash
 - Support the `HAVING()` function in DAG requests

14.5.3.3 Improvements

- TiDB
 - Announce the general availability (GA) of the Stale Read feature
 - Avoid allocation for `paramMarker` to speed up data insertion [#26076](#)
 - Support the stable result mode to make the query results more stable [#25995](#)
 - Support pushing down the built-in function `json_unquote()` to TiKV [#26265](#)
 - Support retrying MPP queries [#26480](#)
 - Change the `LOCK` record into the `PUT` record for the index keys using `point get` or `batch point get` for `UPDATE` reads [#26225](#)
 - Forbid creating views from stale queries [#26200](#)
 - Thoroughly push down the `COUNT(DISTINCT)` aggregation function in the MPP mode [#26194](#)

- Check the availability of TiFlash before launching MPP queries [#26192](#)
- Do not allow setting the read timestamp to a future time [#25763](#)
- Print log warnings when aggregation functions cannot be pushed down in EXPLAIN statements [#25737](#)
- Add the `statements_summary_evicted` table to record the evicted count information of a cluster [#25587](#)
- Improve the MySQL compatibility of the built-in function `str_to_date` for the format specifiers `%b/%M/%r/%T` [#25768](#)
- TiKV
 - Make the prewrite requests as idempotent as possible to reduce the chance of undetermined errors [#10586](#)
 - Prevent the risk of stack overflow when handling many expired commands [#10502](#)
 - Avoid excessive commit request retrying by not using the Stale Read request's `start_ts` to update `max_ts` [#10451](#)
 - Handle read ready and write ready separately to reduce read latency [#10592](#)
 - Reduce the impact on data import speed when the I/O rate limiting is enabled [#10390](#)
 - Improve the load balance between Raft gRPC connections [#10495](#)
- Tools
 - TiCDC
 - * Remove `file sorter` [#2327](#)
 - * Improve the error message returned when a PD endpoint misses the certificate [#1973](#)
 - TiDB Lightning
 - * Add a retry mechanism for restoring schemas [#1294](#)
 - Dumpling
 - * Always split tables using `_tidb_rowid` when the upstream is a TiDB v3.x cluster, which helps reduce TiDB's memory usage [#295](#)
 - * Reduce the frequency of accessing the database metadata to improve Dumpling's performance and stability [#315](#)

14.5.3.4 Bug fixes

- TiDB
 - Fix the data loss issue that might occur when changing the column type with `tidb_enable_amend_pessimistic_txn=on` [#26203](#)
 - Fix the issue that the behavior of the `last_day` function is incompatible in the SQL mode [#26001](#)

- Fix the panic issue that might occur when `LIMIT` is on top of window functions [#25344](#)
 - Fix the issue that committing pessimistic transactions might cause write conflict [#25964](#)
 - Fix the issue that the result of index join in correlated subqueries is wrong [#25799](#)
 - Fix a bug that the successfully committed optimistic transactions might report commit errors [#10468](#)
 - Fix the issue that an incorrect result is returned when using merge join on the `SET` type column [#25669](#)
 - Fix a bug that the index keys in a pessimistic transaction might be repeatedly committed [#26359](#)
 - Fix the risk of integer overflow when the optimizer is locating partitions [#26227](#)
 - Fix the issue that invalid values might be written when casting `DATE` to timestamp [#26292](#)
 - Fix the issue that the Coprocessor Cache metrics are not displayed on Grafana [#26338](#)
 - Fix the issue of annoying logs caused by telemetry [#25760](#) [#25785](#)
 - Fix a bug on the query range of prefix index [#26029](#)
 - Fix the issue that concurrently truncating the same partition hangs DDL executions [#26229](#)
 - Fix the issue of duplicate `ENUM` items [#25955](#)
 - Fix a bug that the CTE iterator is not correctly closed [#26112](#)
 - Fix the issue that the `LOAD DATA` statement might abnormally import non-utf8 data [#25979](#)
 - Fix the panic issue that might occur when using the window function on the unsigned integer columns [#25956](#)
 - Fix the issue that TiDB might panic when resolving async commit locks [#25778](#)
 - Fix the issue that Stale Read is not fully compatible with the `PREPARE` statements [#25800](#)
 - Fix the issue that the ODBC-styled constant (for example, `{d '2020-01-01'}`) cannot be used as the expression [#25531](#)
 - Fix an error that occurs when running TiDB alone [#25555](#)
- TiKV
 - Fix the issue that the duration calculation might panic on certain platforms [#10569](#)
 - Fix the issue that Load Base Split mistakenly uses the unencoded keys of `batch_get_command` [#10542](#)
 - Fix the issue that changing the `resolved-ts.advance-ts-interval` configuration online cannot take effect immediately [#10426](#)
 - Fix the issue of follower metadata corruption in rare cases with more than 4 replicas [#10225](#)
 - Fix the panic issue that occurs when building a snapshot twice if encryption is enabled [#9786](#) [#10407](#)

- Fix the wrong `tikv_raftstore_hibernated_peer_state` metric [#10330](#)
- Fix the wrong arguments type of the `json_unquote()` function in the coprocessor [#10176](#)
- Fix a bug that the index keys in a pessimistic transaction might be repeatedly committed [#10468](#)
- Fix the issue that the `ReadIndex` request returns stale result right after the leader is transferred [#9351](#)
- PD
 - Fix the issue the expected scheduling cannot be generated when the conflict occurs due to multiple schedulers running at the same time [#3807](#) [#3778](#)
 - Fix the issue that the scheduler might appear again even if the scheduler is already deleted [#2572](#)
- TiFlash
 - Fix the potential panic issue that occurs when running table scan tasks
 - Fix a bug that TiFlash raises the error about `duplicated region` when handling DAQ requests
 - Fix the panic issue that occurs when the read load is heavy
 - Fix the potential panic issue that occurs when executing the `DateFormat` function
 - Fix the potential memory leak issue that occurs when executing MPP tasks
 - Fix the issue of unexpected results when executing the aggregation functions `COUNT` or `COUNT DISTINCT`
 - Fix a potential bug that TiFlash cannot restore data when deployed on multiple disks
 - Fix the issue that TiDB Dashboard cannot display the disk information of TiFlash correctly
 - Fix the potential panic issue that occurs when deconstructing `SharedQueryBlockInputStream` ↵
 - Fix the potential panic issue that occurs when deconstructing `MPPTask`
 - Fix the potential issue of data inconsistency after synchronizing data via snapshot
- Tools
 - TiCDC
 - * Fix the support for the new collation feature [#2301](#)
 - * Fix the issue that an unsynchronized access to a shared map at runtime might cause panic [#2300](#)
 - * Fix the potential DDL loss issue that occurs when the owner crashes while executing the DDL statement [#2290](#)
 - * Fix the issue of trying to resolve locks in TiDB prematurely [#2188](#)
 - * Fix a bug that might cause data loss if a TiCDC node is killed immediately after a table migration [#2033](#)

- * Fix the handling logic of `changefeed update` on `--sort-dir` and `--start-
→ ts #1921`
- Backup & Restore (BR)
 - * Fix the issue that the size of the data to restore is incorrectly calculated
[#1270](#)
 - * Fix the issue of missed DDL events that occurs when restoring from cdclog
[#870](#)
- TiDB Lightning
 - * Fix the issue that TiDB fails to parse the `DECIMAL` type data in Parquet files
[#1275](#)
 - * Fix the issue of integer overflow when calculating key intervals [#1291](#) [#1290](#)

14.5.4 TiDB 5.1 Release Notes

Release date: June 24, 2021

TiDB version: 5.1.0

In v5.1, the key new features or improvements are as follows:

- Support the Common Table Expression (CTE) feature of MySQL 8.0 to improve the readability and execution efficiency of SQL statements.
- Support changing column types online to improve code development flexibility.
- Introduce a new statistics type to improve query stability, which is enabled as an experimental feature by default.
- Support the dynamic privilege feature of MySQL 8.0 to implement more fine-grained control over certain operations.
- Support directly reading data from the local replica using the Stale Read feature to reduce read latency and improve query performance (Experimental).
- Add the Lock View feature to facilitate database administrators (DBAs) to observe transaction locking events and troubleshoot deadlock problems (Experimental).
- Add TiKV write rate limiter for background tasks to ensure that the latency of read and write requests is stable.

14.5.4.1 Compatibility changes

Note:

When upgrading from an earlier TiDB version to v5.1, if you want to know the compatibility change notes of all intermediate versions, you can check the [Release Notes](#) for the corresponding version.

14.5.4.1.1 System variables

Variable name	Change type	Description
<code>cte_max_recursion_depth</code>	Newly added →	Controls the maximum recursion depth in Common Table Expressions.
<code>init_connect</code>	Newly added →	Controls the initial connection to a TiDB server.
<code>tidb_analyze_new</code>	Newly added →	Controls how TiDB collects statistics. The default value of this variable is 2. This is an experimental feature.
<code>tidb_enable_enhanced_security</code>	Newly added →	Indicates whether the TiDB server you are connected to has the Security Enhanced Mode (SEM) enabled. This variable setting cannot be changed without restarting the TiDB server.

Variable name	Change type	Description
<code>tidb_enforce_Npp</code>	Newly added ↪	Controls whether to ignore the optimizer's cost estimation and to forcibly use the MPP mode for query execution. The data type of this variable is BOOL and the default value is <code>false</code> .
<code>tidb_partition_pruning_mode</code>	Newly added ↪	Specifies whether to enable dynamic pruning mode for partitioned tables. This feature is experimental. The default value of this variable is <code>static</code> , which means the dynamic pruning mode for partitioned tables is disabled by default.

14.5.4.1.2 Configuration file parameters

Configuration file	Configuration item	Change type	Description
TiDB configuration file	<code>security.</code> ↳ <code>enable-</code> ↳ <code>sem</code>	Newly added	Controls whether to enable the Security Enhanced Mode (SEM). The default value of this configuration item is <code>false</code> , which means the SEM is disabled.
TiDB configuration file	<code>performance</code> ↳ <code>.</code> ↳ <code>committer</code> ↳ <code>-</code> ↳ <code>concurrency</code> ↳	Modified	Controls the concurrency number for requests related to commit operations in the commit phase of a single transaction. The default value is changed from 16 to 128.

Configuration file	Configuration item	Change type	Description
TiDB configuration file	<code>performance</code> ↳ <code>.tcp-no-delay</code>	Newly added	Determines whether to enable TCP_NODELAY at the TCP layer. The default value is <code>true</code> , which means TCP_NODELAY is enabled.
TiDB configuration file	<code>performance</code> ↳ <code>.</code> ↳ <code>enforce-mpp</code>	Newly added	Controls whether TiDB ignores cost estimates of Optimizer at the instance level and enforces the MPP mode. The default value is <code>false</code> . This configuration item controls the initial value of the system variable <code>tidb_enforce_mpp</code> ↳ <code>.</code>

Configuration file	Configuration item	Change type	Description
TiDB configuration file	pessimistic ↳ -txn. ↳ deadlock ↳ - ↳ history ↳ - ↳ capacity ↳	Newly added	Sets the maximum number of deadlock events that can be recorded in the INFORMATION_SCHEMA . ↳ . ↳ DEADLOCKS ↳ table of a single TiDB server. The default value is 10.
TiKV configuration file	abort-on- ↳ panic	Newly added	Sets whether the abort process allows the system to generate core dump files when TiKV panics. The default value is false , which means it is not allowed to generate core dump files.

Configuration file	Configuration item	Change type	Description
TiKV configuration file	<code>hibernate- ↵ regions</code>	Modified	The default value is changed from <code>false</code> to <code>true</code> . If a Region is idle for a long time, it is automatically set as hibernated.
TiKV configuration file	<code>old-value- ↵ cache- ↵ memory- ↵ quota</code>	Newly added	Sets the upper limit of memory usage by TiCDC old values. The default value is <code>512MB</code> .
TiKV configuration file	<code>sink- ↵ memory- ↵ quota</code>	Newly added	Sets the upper limit of memory usage by TiCDC data change events. The default value is <code>512MB</code> .

Configuration file	Configuration item	Change type	Description
TiKV configuration file	<code>incremental</code> ↪ <code>-scan-</code> ↪ <code>threads</code>	Newly added	Sets the number of threads for the task of incrementally scanning historical data. The default value is 4, which means there are four threads for the task.
TiKV configuration file	<code>incremental</code> ↪ <code>-scan-</code> ↪ <code>concurrency</code> ↪	Newly added	Sets the maximum number of concurrent executions for the tasks of incrementally scanning historical data. The default value is 6, which means that six tasks can be concurrently executed at most.

Configuration file	Configuration item	Change type	Description
TiKV configuration file	<code>soft-</code> ↳ <code>pending</code> ↳ <code>-</code> ↳ <code>compaction</code> ↳ <code>-bytes-</code> ↳ <code>limit</code>	Modified	The soft limit on the pending compaction bytes. The default value is changed from "64GB" to "192GB".
TiKV configuration file	<code>storage.io</code> ↳ <code>-rate-</code> ↳ <code>limit</code>	Newly added	Controls the I/O rate of TiKV writes. The default value of <code>storage.</code> is "0MB". ↳ <code>io-rate</code> ↳ <code>-limit.</code> ↳ <code>max-</code> ↳ <code>bytes-</code> ↳ <code>per-sec</code>
TiKV configuration file	<code>resolved-</code> ↳ <code>ts.</code> ↳ <code>enable</code>	Newly added	Determines whether to maintain the <code>resolved-</code> ↳ <code>ts</code> for all Region leaders. The default value is <code>true</code> .
TiKV configuration file	<code>resolved-</code> ↳ <code>ts.</code> ↳ <code>advance</code> ↳ <code>-ts-</code> ↳ <code>interval</code> ↳	Newly added	The interval at which the <code>resolved-</code> ↳ <code>ts</code> is forwarded. The default value is "1s". You can change the value dynamically.

Configuration file	Configuration item	Change type	Description
TiKV configuration file	resolved- ↵ ts.scan ↵ -lock- ↵ pool- ↵ size	Newly added	The number of threads that TiKV uses to scan the MVCC (multi-version concurrency control) lock data when initializing the resolved- ↵ ts . The default value is 2.

14.5.4.1.3 Others

- Before the upgrade, check the value of the TiDB configuration **feedback-probability** ↵ . If the value is not 0, the “panic in the recoverable goroutine” error will occur after the upgrade, but this error does not affect the upgrade.
- Upgrade the Go compiler version of TiDB from go1.13.7 to go1.16.4, which improves the TiDB performance. If you are a TiDB developer, upgrade your Go compiler version to ensure a smooth compilation.
- Avoid creating tables with clustered indexes in the cluster that uses TiDB Binlog during the TiDB rolling upgrade.
- Avoid executing statements like `alter table ... modify column` or `alter table ... change column` during the TiDB rolling upgrade.
- Since v5.1, setting the replica of system tables, when building TiFlash replicas for each table, is no longer supported. Before upgrading the cluster, you need to clear the relevant system table replicas; otherwise, the upgrade will fail.
- Deprecate the `--sort-dir` parameter in the `cdb cli changefeed` command of TiCDC. Instead, you can set `--sort-dir` in the `cdb server` command. #1795
- After upgrading to TiDB 5.1, if TiDB returns the “function READ ONLY has only noop implementation” error, you can let TiDB ignore this error by setting the value of `tidb_enable_noop_functions` to ON. This is because the `read_only` variable in MySQL does not yet take effect in TiDB (which is a ‘noop’ behavior in TiDB). Therefore, even if this variable is set in TiDB, you can still write data into the TiDB cluster.

14.5.4.2 New features

14.5.4.2.1 SQL

- Support the Common Table Expression (CTE) feature of MySQL 8.0.

This feature empowers TiDB with the capability of querying hierarchical data recursively or non-recursively and meets the needs of using tree queries to implement application logics in multiple sectors such as human resources, manufacturing, financial markets, and education.

In TiDB, you can apply the `WITH` statement to use Common Table Expressions. [User document](#), #17472

- Support the dynamic privilege feature of MySQL 8.0.

Dynamic privileges are used to limit the `SUPER` privilege and provide TiDB with more flexible privilege configuration for more fine-grained access control. For example, you can use dynamic privileges to create a user account that can only perform `BACKUP` and `RESTORE` operations.

The supported dynamic privileges are as follows:

- `BACKUP_ADMIN`
- `RESTORE_ADMIN`
- `ROLE_ADMIN`
- `CONNECTION_ADMIN`
- `SYSTEM_VARIABLES_ADMIN`

You can also use plugins to add new privileges. To check out all supported privileges, execute the `SHOW PRIVILEGES` statement. [User document](#)

- Add a new configuration item for the Security Enhanced Mode (SEM), which divides the TiDB administrator privileges in a finer-grained way.

The Security Enhanced Mode is disabled by default. To enable it, see the [user document](#).

- Enhance the capability of changing column types online. Support changing the column type online using the `ALTER TABLE` statement, including but not limited to:

- Changing `VARCHAR` to `BIGINT`
- Modifying the `DECIMAL` precision
- Compressing the length of `VARCHAR(10)` to `VARCHAR(5)`

[User document](#)

- Introduce a new SQL syntax `AS OF TIMESTAMP` to perform Stale Read, a new experimental feature used to read historical data from a specified point in time or from a specified time range.

[User document](#), #21094

The examples of `AS OF TIMESTAMP` are as follows:

```
SELECT * FROM t AS OF TIMESTAMP '2020-09-06 00:00:00';
START TRANSACTION READ ONLY AS OF TIMESTAMP '2020-09-06 00:00:00';
SET TRANSACTION READ ONLY as of timestamp '2020-09-06 00:00:00';
```

- Introduce a new statistics type `tidb_analyze_version = 2` (Experimental).
`tidb_analyze_version` is set to 2 by default, which avoids the large errors that might occur in the large data volume caused by hash conflicts in Version 1 and maintains the estimation accuracy in most scenarios.

[User document](#)

14.5.4.2.2 Transaction

- Support the Lock View feature (Experimental)

The Lock View feature provides more information about lock conflicts and lock waits of pessimistic locks, which helps DBAs to observe transaction locking conditions and troubleshoot deadlock problems. [#24199](#)

User document:

- View the pessimistic locks and other locks that currently occur on all TiKV nodes in the clusters: [DATA_LOCK_WAIT](#)
- View several deadlock errors that recently occurred on the TiDB nodes: [DEADLOCKS](#)
- View the transaction information executed currently on the TiDB nodes: [TIDB_TRX](#)

14.5.4.2.3 Performance

- Stale read of data replicas (Experimental)

Read local replicas data directly to reduce read latency and improve the query performance

[User document](#), [#21094](#)

- Enable the Hibernate Region feature by default.

If a Region is in an inactive state for a long time, it is automatically set to a silent state, which reduces the system overhead of the heartbeat information between the Leader and the Follower.

[User document](#), [#10266](#)

14.5.4.2.4 Stability

- Solve the replication stability issue of TiCDC
 - Improve TiCDC memory usage to avoid OOM in the following scenarios
 - If large amounts of data is accumulated during the replication interruption, exceeding 1TB, the re-replication causes OOM problems.
 - Large amounts of data writes cause OOM problems in TiCDC.
 - Reduce the possibility of TiCDC replication interruption in the following scenarios:
 - project #11
 - * Replication interruption when the network is unstable
 - * Replication interruption when some TiKV/PD/TiCDC nodes are down

- TiFlash storage memory control

Optimize the speed and memory usage of Region snapshot generation and reduce the possibility of OOM

- Add a write rate limiter for TiKV background tasks (TiKV Write Rate Limiter)

To ensure the duration stability of read and write requests, TiKV Write Rate Limiter smoothes the write traffic of TiKV background tasks such as GC and Compaction. The default value of TiKV background task write rate limiter is “0MB”. It is recommended to set this value to the optimal I/O bandwidth of the disk, such as the maximum I/O bandwidth specified by the cloud disk manufacturer.

[User document](#), #9156

- Solve scheduling stability issues when multiple scaling tasks are performed at the same time

14.5.4.2.5 Telemetry

TiDB adds the running status of TiDB cluster requests in telemetry, including execution status, failure status, etc.

To learn more about the information and how to disable this behavior, refer to [Telemetry](#).

14.5.4.3 Improvements

- TiDB
 - Support the built-in function VITNESS_HASH() [#23915](#)
 - Support pushing down data of the enumerated type to TiKV to improve performance when using enumerated types in WHERE clauses [#23619](#)

- Optimize the calculation of Window Function to solve TiDB OOM problems when paging data with `ROW_NUMBER()` [#23807](#)
- Optimize the calculation of `UNION ALL` to solve the TiDB OOM problems when using `UNION ALL` to join a large number of `SELECT` statements [#21441](#)
- Optimize the dynamic pruning mode of partitioned tables to improve performance and stability [#24150](#)
- Fix the `Region is Unavailable` issue that occurs in multiple scenarios project[#62](#)
- Fix multiple `Region is Unavailable` issues that might occur in frequent scheduling situations
- Fix `Region is Unavailable` issue that might occur in some high stress write situations
- Avoid frequently reading the `mysql.stats_histograms` table if the cached statistics is up-to-date to avoid high CPU usage [#24317](#)
- TiKV
 - Use `zstd` to compress Region snapshots, preventing large space differences between nodes in case of heavy scheduling or scaling [#10005](#)
 - Solve OOM issues in multiple cases [#10183](#)
 - * Add memory usage tracking for each module
 - * Solve the OOM issue caused by oversized Raft entries cache
 - * Solve the OOM issue caused by stacked GC tasks
 - * Solve the OOM issue caused by fetching too many Raft entries from the Raft log to memory at one time
 - Split Regions more evenly to mitigate the issue that the growth of Region size exceeds the splitting speed when there are hotspot writes [#9785](#)
- TiFlash
 - Support `Union All`, `TopN`, and `Limit` functions
 - Support the Cartesian product including left outer join and semi anti join in MPP mode
 - Optimize lock operations to avoid that running DDL statements and read operations are blocked by each other
 - Optimize cleanup of expired data by TiFlash
 - Support further filtering of query filters on `timestamp` columns at the TiFlash storage level
 - Improve the startup and scalability speed of TiFlash when a large number of tables are in a cluster
 - Improve TiFlash compatibility when running on unknown CPUs
- PD
 - Avoid unexpected statistics after adding the `scatter region` scheduler [#3602](#)

- Solve multiple scheduling issues in the scaling process
 - * Optimize the generation process of replica snapshots to solve slow scheduling issues during scaling [#3563](#) [#10059](#) [#10001](#)
 - * Solve slow scheduling issues caused by heartbeat pressure due to traffic changes [#3693](#) [#3739](#) [#3728](#) [#3751](#)
 - * Reduce the space discrepancies of large clusters due to scheduling, and optimize the scheduling formula to prevent the bursting issue (which is similar to heterogeneous space clusters) caused by large compression rate discrepancies [#3592](#) [#10005](#)
- Tools
 - Backup & Restore (BR)
 - * Support backing up and restoring system tables in the mysql schema [#1143](#) [#1078](#)
 - * Support the S3-compatible storage that is based on the virtual-host addressing mode [#10243](#)
 - * Optimize the format of backupmeta to reduce memory usage [#1171](#)
 - TiCDC
 - * Improve the descriptions of some log messages to be clearer and more useful for diagnosing problems [#1759](#)
 - * Support the back pressure feature to allow the TiCDC scanning speed to sense the downstream processing capacity [#10151](#)
 - * Reduce memory usage when TiCDC performs the initial scan [#10133](#)
 - * Improve the cache hit rate for the TiCDC Old Value in pessimistic transactions [#10089](#)
 - Dumpling
 - * Improve the logic of exporting data from TiDB v4.0 to avoid TiDB becoming out of memory (OOM) [#273](#)
 - * Fix the issue that no error is output when a backup operation fails [#280](#)
 - TiDB Lightning
 - * Improve data importing speed. The optimization results show that the speed of importing TPC-C data is increased by 30%, and the speed of importing large tables (2TB+) with more indexes (5 indexes) is increased by more than 50%. [#753](#)
 - * Add a pre-check on the data to be imported and also on the target cluster before importing, and report errors to reject the import process if it does not meet the import requirements [#999](#)
 - * Optimize the timing of checkpoint updates on the Local backend to improve performance of restarting from breakpoints [#1080](#)

14.5.4.4 Bug Fixes

- TiDB
 - Fix the issue that the execution result of project elimination might be wrong when the projection result is empty [#23887](#)
 - Fix the issue of wrong query results when a column contains NULL values in some cases [#23891](#)
 - Forbid generating MPP plans when the scan contains virtual columns [#23886](#)
 - Fix the wrong reuse of PointGet and TableDual in Plan Cache [#23187](#) [#23144](#) [#23304](#) [#23290](#)
 - Fix the error that occurs when the optimizer builds the IndexMerge plan for clustered indexes [#23906](#)
 - Fix the type inference of the BIT-type errors [#23832](#)
 - Fix the issue that some optimizer hints do not take effect when the PointGet operator exists [#23570](#)
 - Fix the issue that DDL operations might fail when rolling back due to an error [#23893](#)
 - Fix the issue that the index range of the binary literal constant is incorrectly built [#23672](#)
 - Fix the potential wrong results of the IN clause in some cases [#23889](#)
 - Fix the wrong results of some string functions [#23759](#)
 - Users now need both INSERT and DELETE privileges on a table to perform REPLACE operations [#23909](#)
 - Users now need both INSERT and DELETE privileges on a table to perform REPLACE operations [#24070](#)
 - Fix the wrong TableDual plans caused by incorrectly comparing binaries and bytes [#23846](#)
 - Fix the panic issue caused by using the prefix index and index join in some cases [#24547](#) [#24716](#) [#24717](#)
 - Fix the issue that the prepared plan cache of point get is incorrectly used by the point get statement in the transaction [#24741](#)
 - Fix the issue of writing the wrong prefix index value when the collation is ascii_bin or latin1_bin [#24569](#)
 - Fix the issue that the ongoing transaction might be interrupted by the GC worker [#24591](#)
 - Fix a bug that the point query might get wrong on the clustered index when new-collation is enabled but new-row-format is disabled [#24541](#)
 - Refactor the conversion of partition keys for shuffle hash join [#24490](#)
 - Fix the panic issue that occurs when building the plan for queries that contain the HAVING clause [#24045](#)
 - Fix the issue that the column pruning improvement causes the Apply and Join operators' results to go wrong [#23887](#)
 - Fix a bug that the primary lock fallen back from async commit cannot be resolved [#24384](#)

- Fix a GC issue of statistics that might cause duplicated fm-sketch records [#24357](#)
- Avoid unnecessary pessimistic rollback when the pessimistic locking receives the `ErrKeyExists` error [#23799](#)
- Fix the issue that numeric literals cannot be recognized when the `sql_mode` contains `ANSI_QUOTES` [#24429](#)
- Forbid statements such as `INSERT INTO table PARTITION (<partitions>)...`
 ↳ `ON DUPLICATE KEY UPDATE` to read data from non-listed partitions [#24746](#)
- Fix the potential `index out of range` error when a SQL statement contains both `GROUP BY` and `UNION` [#24281](#)
- Fix the issue that the `CONCAT` function incorrectly handles the collation [#24296](#)
- Fix the issue that the `collation_server` global variable does not take effect in new sessions [#24156](#)

- TiKV

- Fix the issue that the coprocessor fails to properly handle the signed or unsigned integer types in the `IN` expression [#9821](#)
- Fix the issue of many empty Regions after batch ingesting SST files [#964](#)
- Fix a bug that TiKV cannot start up after the file dictionary file is damaged [#9886](#)
- Fix a TiCDC OOM issue caused by reading old values [#9996](#) [#9981](#)
- Fix the issue of empty value in the secondary index for the clustered primary key column when collation is `latin1_bin` [#24548](#)
- Add the `abort-on-panic` configuration, which allows TiKV to generate the core dump file when panic occurs. Users still need to correctly configure the environment to enable core dump [#10216](#)
- Fix the performance regression issue of `point get` queries that occurs when TiKV is not busy [#10046](#)

- PD

- Fix the issue that the PD Leader re-election is slow when there are many stores [#3697](#)
- Fix the panic issue that occurs when removing the evict leader scheduler from a non-existent store [#3660](#)
- Fix the issue that the statistics are not updated after offline peers are merged [#3611](#)

- TiFlash

- Fix the issue of incorrect results when casting the time type to the integer type
- Fix a bug that the `receiver` cannot find corresponding tasks within 10 seconds
- Fix the issue that there might be invalid iterators in `cancelMPPQuery`
- Fix a bug that the behavior of the `bitwise` operator is different from that of TiDB

- Fix the alert issue caused by overlapping ranges when using the `prefix` key
- Fix the issue of incorrect results when casting the string type to the integer type
- Fix the issue that consecutive and fast writes might make TiFlash out of memory
- Fix the potential issue that the exception of null pointer might be raised during the table GC
- Fix the TiFlash panic issue that occurs when writing data to dropped tables
- Fix the issue that TiFlash might panic during BR restore
- Fix the issue of incorrect results when cloning shared delta index concurrently
- Fix the potential panic that occurs when the Compaction Filter feature is enabled
- Fix the issue that TiFlash cannot resolve the lock fallen back from async commit
- Fix the issue of incorrect results returned when the casted result of the `TIMEZONE` type contains the `TIMESTAMP` type
- Fix the TiFlash panic issue that occurs during Segment Split
- Tools
 - TiDB Lightning
 - * Fix the issue of TiDB Lightning panic that occurs when generating KV data [#1127](#)
 - * Fix a bug that the batch split Region fails due to the total key size exceeding the raft entry limit during the data import [#969](#)
 - * Fix the issue that when importing CSV files, if the last line of the file does not contain line break characters (`\r\n`), an error will be reported [#1133](#)
 - * Fix the issue that if a table to be imported includes an auto-increment column of the double type, the `auto_increment` value becomes abnormal [#1178](#)
 - Backup & Restore (BR)
 - * Fix the issue of backup interruption caused by the failure of a few TiKV nodes [#980](#)
 - TiCDC
 - * Fix the concurrency issue in Unified Sorter and filter the unhelpful error messages [#1678](#)
 - * Fix a bug that the creation of redundant directories might interrupt the replication with MinIO [#1463](#)
 - * Set the default value of the `explicit_defaults_for_timestamp` session variable to ON to make the MySQL 5.7 downstream keep the same behavior with the upstream TiDB [#1585](#)
 - * Fix the issue that the incorrect handling of `io.EOF` might cause replication interruption [#1633](#)
 - * Correct the TiKV CDC endpoint CPU metric in the TiCDC dashboard [#1645](#)
 - * Increase `defaultBufferChanSize` to avoid replication blocking in some cases [#1259](#)
 - * Fix the issue that the time zone information is lost in the Avro output [#1712](#)
 - * Support cleaning up stale temporary files in Unified Sorter and forbid sharing the `sort-dir` directory [#1742](#)

- * Fix a deadlock bug in the KV client that occurs when many stale Regions exist [#1599](#)
- * Fix the wrong help information in the `--cert-allowed-cn` flag [#1697](#)
- * Revert the update for `explicit_defaults_for_timestamp` which requires the SUPER privilege when replicating data to MySQL [#1750](#)
- * Support the sink flow control to reduce the risk of memory overflow [#1840](#)
- * Fix a bug that the replication task might stop when moving a table [#1828](#)
- * Fix the issue that the TiKV GC safe point is blocked due to the stagnation of TiCDC changefeed checkpoint [#1759](#)

14.6 v5.0

14.6.1 TiDB 5.0.5 Release Note

Release date: December 3, 2021

TiDB version: 5.0.5

14.6.1.1 Bug fix

- TiKV
 - Fix the issue that the `GcKeys` task does not work when it is called by multiple keys. Caused by this issue, compaction filer GC might not drop the MVCC deletion information. [#11217](#)

14.6.2 TiDB 5.0.4 Release Notes

Release Date: September 27, 2021

TiDB version: 5.0.4

14.6.2.1 Compatibility changes

- TiDB
 - Fix the issue that executing `SHOW VARIABLES` in a new session is slow. This fix reverts some changes made in [#19341](#) and might cause compatibility issues. [#24326](#)
 - Change the default value of the `tidb_stmt_summary_max_stmt_count` variable from 200 to 3000 [#25873](#)
 - The following bug fixes change execution results, which might cause upgrade incompatibilities:

- * Fix the issue that TiDB returns wrong result when the children of `UNION` contain the `NULL` value [#26559](#)
- * Fix the issue that `greatest(datetime)union null` returns empty string [#26532](#)
- * Fix the issue that the behavior of the `last_day` function is incompatible in SQL mode [#26000](#)
- * Fix the issue that the `having` clause might not work correctly [#26496](#)
- * Fix the wrong execution results that occur when the collations around the `between` expression are different [#27146](#)
- * Fix the wrong execution results that occur when the column in the `group_concat` function has a non-bin collation [#27429](#)
- * Fix an issue that using a `count(distinct)` expression on multiple columns returns wrong result when the new collation is enabled [#27091](#)
- * Fix the result wrong that occurs when the argument of the `extract` function is a negative duration [#27236](#)
- * Fix the issue that inserting an invalid date does not report an error when the `SQL_MODE` is '`STRICT_TRANS_TABLES`' [#26762](#)
- * Fix the issue that using an invalid default date does not report an error when the `SQL_MODE` is '`NO_ZERO_IN_DATE`' [#26766](#)
- * Fix a bug on the query range of prefix index [#26029](#)
- * Fix the issue that the `LOAD DATA` statement might abnormally import non-`utf8` data [#25979](#)
- * Fix the issue that `insert ignore on duplicate update` might insert wrong data when the secondary index has the same column with the primary key [#25809](#)
- * Fix the issue that `insert ignore duplicate update` might insert wrong data when a partitioned table has a clustered index [#25846](#)
- * Fix the issue that the query result might be wrong when the key is the `ENUM` type in point get or batch point get [#24562](#)
- * Fix the wrong result that occurs when dividing a `BIT`-type value [#23479](#)
- * Fix the issue that the results of prepared statements and direct queries might be inconsistent [#22949](#)
- * Fix the issue that the query result might be wrong when a `YEAR` type is compared with a string or an integer type [#23262](#)

14.6.2.2 Feature enhancements

- TiDB
 - Support setting `tidb_enforce_mpp=1` to ignore the optimizer estimation and forcibly use the MPP mode [#26382](#)
- TiKV
 - Support changing TiCDC configurations dynamically [#10645](#)

- PD
 - Add OIDC-based SSO support for TiDB Dashboard [#3884](#)
- TiFlash
 - Support the HAVING() function in DAG requests
 - Support the DATE() function
 - Add Grafana panels for write throughput per instance

14.6.2.3 Improvements

- TiDB
 - Trigger auto-analyze based on the histogram row count [#24237](#)
 - Stop sending requests to a TiFlash node for a period if the node has failed and restarted before [#26757](#)
 - Increase the `split_region` upper limit to make `split_table` and `presplit` more stable [#26657](#)
 - Support retry for MPP queries [#26483](#)
 - Check the availability of TiFlash before launching MPP queries [#1807](#)
 - Support the stable result mode to make the query result more stable [#26084](#)
 - Support the MySQL system variable `init_connect` and its associated features [#18894](#)
 - Thoroughly push down the `COUNT(DISTINCT)` aggregation function in the MPP mode [#25861](#)
 - Print log warnings when the aggregation function cannot be pushed down in `EXPLAIN` statements [#25736](#)
 - Add error labels for `TiFlashQueryTotalCounter` in Grafana dashboard [#25327](#)
 - Support getting the MVCC data of a clustered index table through a secondary index by HTTP API [#24209](#)
 - Optimize the memory allocation of `prepared` statement in parser [#24371](#)
- TiKV
 - Handle read ready and write ready separately to reduce read latency [#10475](#)
 - Reduce the size of Resolved TS messages to save network bandwidth [#2448](#)
 - Drop log instead of blocking threads when the slogger thread is overloaded and the queue is filled up [#10841](#)
 - Make the slow log of TiKV coprocessor only consider the time spent on processing requests [#10841](#)
 - Make prewrite as idempotent as possible to reduce the chance of undetermined errors [#10587](#)
 - Avoid the false “GC can not work” alert under low write flow [#10662](#)

- Make the database to be restored always match the original cluster size during backup. [#10643](#)
- Ensure that the panic output is flushed to the log [#9955](#)
- PD
 - Improve the performance of synchronizing Region information between PDs [#3993](#)
- Tools
 - Dumpling
 - * Support backing up MySQL-compatible databases that do not support the `START TRANSACTION ... WITH CONSISTENT SNAPSHOT` or the `SHOW CREATE ↵ TABLE` syntax [#309](#)
 - TiCDC
 - * Optimize memory management when Unified Sorter is using memory to sort [#2553](#)
 - * Prohibit operating TiCDC clusters across major or minor versions [#2598](#)
 - * Reduce the goroutine usage when a table's Regions are all transferred away from a TiKV node [#2284](#)
 - * Remove `file sorter` [#2326](#)
 - * Always pull the old values from TiKV and the output is adjusted according to `enable-old-value` [#2301](#)
 - * Improve the error message returned when a PD endpoint misses the certificate [#1973](#)
 - * Optimize workerpool for fewer goroutines when concurrency is high [#2211](#)
 - * Add a global gRPC connection pool and share gRPC connections among KV clients [#2533](#)

14.6.2.4 Bug Fixes

- TiDB
 - Fix the issue that TiDB might panic when querying a partitioned table and the partition key has the `IS NULL` condition [#23802](#)
 - Fix the issue that the overflow check of the `FLOAT64` type is different with that of MySQL [#23897](#)
 - Fix the wrong character set and collation for the `case when` expression [#26662](#)
 - Fix the issue that committing pessimistic transactions might cause write conflicts [#25964](#)
 - Fix a bug that the index keys in a pessimistic transaction might be repeatedly committed [#26359](#) [#10600](#)

- Fix the issue that TiDB might panic when resolving the async commit locks [#25778](#)
 - Fix a bug that a column might not be found when using INDEX MERGE [#25045](#)
 - Fix a bug that ALTER USER REQUIRE SSL clears users' authentication_string [#25225](#)
 - Fix a bug that the value of the `tidb_gc_scan_lock_mode` global variable on a new cluster shows “PHYSICAL” instead of the actual default mode “LEGACY” [#25100](#)
 - Fix the bug that the `TIKV_REGION_PEERS` system table does not show the correct DOWN status [#24879](#)
 - Fix the issue of memory leaks that occurs when HTTP API is used [#24649](#)
 - Fix the issue that views do not support DEFINER [#24414](#)
 - Fix the issue that `tidb-server --help` exits with the code 2 [#24046](#)
 - Fix the issue that setting the global variable `dml_batch_size` does not take effect [#24709](#)
 - Fix the issue that using `read_from_storage` and partitioned table at the same time causes an error [#20372](#)
 - Fix the issue that TiDB panics when executing the projection operator [#24264](#)
 - Fix the issue that statistics might cause queries to panic [#24061](#)
 - Fix the issue that using the `approx_percentile` function on a BIT column might panic [#23662](#)
 - Fix the issue that the metrics on the **Coprocessor Cache** panel in Grafana are wrong [#26338](#)
 - Fix the issue that concurrently truncating the same partition causes DDL statements to stuck [#26229](#)
 - Fix the issue of wrong query results that occurs when the session variable is used as the GROUP BY item [#27106](#)
 - Fix the wrong implicit conversion between VARCHAR and timestamp when joining tables [#25902](#)
 - Fix the wrong results in associated subquery statements [#27233](#)
- TiKV
 - Fix the potential disk full issue caused by corrupted snapshot files [#10813](#)
 - Fix the TiKV panic issue that occurs when upgrading from a pre-5.0 version with Titan enabled [#10843](#)
 - Fix the issue that TiKV of a newer version cannot be rolled back to v5.0.x [#10843](#)
 - Fix the TiKV panic issue that occurs when upgrading from a pre-5.0 version to a 5.0 version or later. If a cluster was upgraded from TiKV v3.x with Titan enabled before the upgrade, this cluster might encounter the issue. [#10774](#)
 - Fix the parsing failure caused by the left pessimistic locks [#26404](#)
 - Fix the panic that occurs when calculating duration on certain platforms [#10571](#)
 - Fix the issue that the keys of `batch_get_command` in Load Base Split are unencoded [#10542](#)
 - PD

- Fix the issue that PD does not fix the down peers in time [#4077](#)
- Fix the issue that the replica count of the default placement rules stays constant after `replication.max-replicas` is updated [#3886](#)
- Fix a bug that PD might panic when scaling out TiKV [#3868](#)
- Fix the scheduling conflict issue that occurs when multiple schedulers are running at same time [#3807](#)
- Fix the issue that the scheduler might appear again even if it has been deleted [#2572](#)
- TiFlash
 - Fix the potential panic issue that occurs when running table scan tasks
 - Fix the potential memory leak issue that occurs when executing MPP tasks
 - Fix a bug that TiFlash raises the `duplicated region` error when handling DAQ requests
 - Fix the issue of unexpected results when executing the aggregation functions `COUNT` or `COUNT DISTINCT`
 - Fix the potential panic issue that occurs when executing MPP tasks
 - Fix a potential bug that TiFlash cannot restore data when deployed on multiple disks
 - Fix the potential panic issue that occurs when deconstructing `SharedQueryBlockInputStream`
 ↪
 - Fix the potential panic issue that occurs when deconstructing `MPPTask`
 - Fix the issue of unexpected results when TiFlash fails to establish MPP connections
 - Fix the potential panic issue that occurs when resolving locks
 - Fix the issue that the store size in metrics is inaccurate under heavy writing
 - Fix a bug of incorrect results that occurs when queries contain filters like `CONSTANT`
 ↪ , <, <=, >, >=, or `COLUMN`
 - Fix the potential issue that TiFlash cannot garbage-collect the delta data after running for a long time
 - Fix a potential bug that metrics display wrong values
 - Fix the potential issue of data inconsistency that occurs when TiFlash is deployed on multiple disks
- Tools
 - Dumpling
 - * Fix the issue that the execution of `show table status` is stuck in MySQL 8.0.3 or a later version [#322](#)
 - TiCDC
 - * Fix the issue of process panic that occurs when encoding the data types such as `mysql.TypeString`, `mysql.TypeVarString`, `mysql.TypeVarchar` into JSON [#2758](#)

- * Fix a data inconsistency issue that occurs because multiple processors might write data to the same table when this table is being re-scheduled [#2417](#)
- * Decrease the gRPC window size to avoid the OOM that occurs when TiCDC captures too many Regions [#2724](#)
- * Fix the error that the gRPC connection is frequently broken when the memory pressure is high [#2202](#)
- * Fix a bug that causes TiCDC to panic on the unsigned TINYINT type [#2648](#)
- * Fix the issue that TiCDC Open Protocol outputs an empty value when inserting a transaction and deleting data of the same row in the upstream [#2612](#)
- * Fix a bug that DDL handling fails when a changefeed starts at the finish TS of a schema change [#2603](#)
- * Fix the issue that irresponsible downstreams interrupt the replication task in old owner until the task times out [#2295](#)
- * Fix a bug in metadata management [#2558](#)
- * Fix the issue of data inconsistency that occurs after the TiCDC owner switch [#2230](#)
- * Fix the issue that outdated capture might appear in the output of the `capture ↳ list` command [#2388](#)
- * Fix the `ErrSchemaStorageTableMiss` error that occurs when the DDL Job duplication is encountered in the integrated test [#2422](#)
- * Fix the bug that a changefeed cannot be removed if the `ErrGCTTLExceeded` error occurs [#2391](#)
- * Fix a bug that replicating large tables to cdclog fails [#1259](#) [#2424](#)
- * Fix the CLI backward compatibility issue [#2373](#)
- * Fix the issue of insecure concurrent access to the map in `SinkManager` [#2299](#)
- * Fix the issue of potential DDL loss when the owner crashes when executing DDL statements [#1260](#)
- * Fix the issue that the lock is resolved immediately after a Region is initialized [#2188](#)
- * Fix the issue of extra partition dispatching that occurs when adding a new partitioned table [#2263](#)
- * Fix the issue that TiCDC keeps warning on removed changefeeds [#2156](#)

14.6.3 TiDB 5.0.3 Release Notes

Release date: July 2, 2021

TiDB version: 5.0.3

14.6.3.1 Compatibility Changes

- TiDB

- After a v4.0 cluster is upgraded to v5.0 or a later version (dev or v5.1), the default value of the `tidb_multi_statement_mode` variable changes from `WARN` to `OFF`

- TiDB is now compatible with MySQL 5.7's noop variable `innodb_default_row_format`
 ↳ . Setting this variable will have no effect. [#23541](#)

14.6.3.2 Feature Enhancements

- Tools
 - TiCDC
 - * Add an HTTP API to get the changefeed information and the health information of the node [#1955](#)
 - * Add the SASL/SCRAM support for the kafka sink [#1942](#)
 - * Make TiCDC support `--data-dir` at the server level [#2070](#)

14.6.3.3 Improvements

- TiDB
 - Support pushing down the `TopN` operator to TiFlash [#25162](#)
 - Support pushing down the built-in function `json_unquote()` to TiKV [#24415](#)
 - Support removing the union branch from the dual table [#25614](#)
 - Support pushing down the built-in function `replace()` to TiFlash [#25565](#)
 - Support pushing down the built-in functions `unix_timestamp()`, `concat()`, `year`
 ↳ `()`, `day()`, `datediff()`, `datesub()`, and `concat_ws()` to TiFlash [#25564](#)
 - Optimize the aggregate operator's cost factor [#25241](#)
 - Support pushing down the `Limit` operator to TiFlash [#25159](#)
 - Support pushing down the built-in function `str_to_date` to TiFlash [#25148](#)
 - Allow the MPP outer join to choose the build table based on the table row count [#25142](#)
 - Support pushing down the built-in functions `left()`, `right()`, and `abs()` to TiFlash [#25133](#)
 - Support pushing down the Broadcast Cartesian join to TiFlash [#25106](#)
 - Support pushing down the `Union All` operator to TiFlash [#25051](#)
 - Support balancing the MPP query workload among different TiFlash nodes based on Regions [#24724](#)
 - Support invalidating stale Regions in the cache after the MPP query is executed [#24432](#)
 - Improve the MySQL compatibility of the built-in function `str_to_date` for the format specifiers `%b/%M/%r/%T` [#25767](#)
- TiKV
 - Limit the TiCDC sink's memory consumption [#10305](#)
 - Add the memory-bounded upper limit for the TiCDC old value cache [#10313](#)

- PD
 - Update TiDB Dashboard to v2021.06.15.1 [#3798](#)
- TiFlash
 - Support casting the STRING type to the DOUBLE type
 - Support the STR_TO_DATE() function
 - Optimize the non-joined data in right outer join using multiple threads
 - Support the Cartesian join
 - Support the LEFT() and RIGHT() functions
 - Support automatically invalidating stale Regions in MPP queries
 - Support the ABS() function
- Tools
 - TiCDC
 - * Refine gRPC's reconnection logic and increase the KV client's throughput
[#1586](#) [#1501](#) [#1682](#) [#1393](#) [#1847](#) [#1905](#) [#1904](#)
 - * Make the sorter I/O errors more user-friendly

14.6.3.4 Bug Fixes

- TiDB
 - Fix the issue that an incorrect result is returned when using merge join on the SET type column [#25669](#)
 - Fix the data corruption issue in the IN expression's arguments [#25591](#)
 - Avoid the sessions of GC being affected by global variables [#24976](#)
 - Fix the panic issue that occurs when using limit in the window function queries [#25344](#)
 - Fix the wrong value returned when querying a partitioned table using Limit [#24636](#)
 - Fix the issue that IFNULL does not correctly take effect on the ENUM or SET type column [#24944](#)
 - Fix the wrong results caused by changing the count in the join subqueries to first_row [#24865](#)
 - Fix the query hang issue that occurs when ParallelApply is used under the TopN operator [#24930](#)
 - Fix the issue that more results than expected are returned when executing SQL statements using multi-column prefix indexes [#24356](#)
 - Fix the issue that the <=> operator cannot correctly take effect [#24477](#)
 - Fix the data race issue of the parallel Apply operator [#23280](#)
 - Fix the issue that the index out of range error is reported when sorting the IndexMerge results of the PartitionUnion operator [#23919](#)

- Fix the issue that setting the `tidb_snapshot` variable to an unexpectedly large value might damage the transaction isolation [#25680](#)
 - Fix the issue that the ODBC-styled constant (for example, `{d '2020-01-01'}`) cannot be used as the expression [#25531](#)
 - Fix the issue that `SELECT DISTINCT` converted to `Batch Get` causes incorrect results [#25320](#)
 - Fix the issue that backing off queries from TiFlash to TiKV cannot be triggered [#23665](#) [#24421](#)
 - Fix the `index-out-of-range` error that occurs when checking `only_full_group_by` ↗ [#23839](#)
 - Fix the issue that the result of index join in correlated subqueries is wrong [#25799](#)
- TiKV
 - Fix the wrong `tikv_raftstore_hibernated_peer_state` metric [#10330](#)
 - Fix the wrong arguments type of the `json_unquote()` function in the coprocessor [#10176](#)
 - Skip clearing callback during graceful shutdown to avoid breaking ACID in some cases [#10353](#) [#10307](#)
 - Fix a bug that the read index is shared for replica reads on a Leader [#10347](#)
 - Fix the wrong function that casts `DOUBLE` to `DOUBLE` [#25200](#)
 - PD
 - Fix the data race issue that occurs when loading TTL configurations after the scheduler is started [#3771](#)
 - Fix a bug that the `is_learner` field of the `TIKV_REGION_PEERS` table in TiDB is incorrect [#3372](#) [#24293](#)
 - Fix the issue that when all TiKV nodes in a zone are offline or down, PD does not schedule replicas to other zones [#3705](#)
 - Fix the issue that PD might get panic after the scatter Region scheduler is added [#3762](#)
 - TiFlash
 - Fix the issue that TiFlash keeps restarting because of the split failure
 - Fix the potential issue that TiFlash cannot delete the delta data
 - Fix a bug that TiFlash adds wrong padding for non-binary characters in the `CAST` function
 - Fix the issue of incorrect results when handling aggregation queries with complex `GROUP BY` columns
 - Fix the TiFlash panic issue that occurs under heavy write pressure
 - Fix the panic that occurs when the right join key is not nullable and the left join key is nullable
 - Fix the potential issue that the `read-index` requests take a long time
 - Fix the panic issue that occurs when the read load is heavy

- Fix the panic issue that might occur when the `Date_Format` function is called with the `STRING` type argument and `NULL` values
- Tools
 - TiCDC
 - * Fix the issue that TiCDC owner exits when refreshing the checkpoint [#1902](#)
 - * Fix a bug that some MySQL connection might leak after MySQL sink meets the error and pauses [#1946](#)
 - * Fix the panic issue that occurs when TiCDC fails to read `/proc/meminfo` [#2024](#)
 - * Reduce TiCDC's runtime memory consumption [#2012](#) [#1958](#)
 - * Fix a bug that might cause TiCDC server panic due to the late calculation of resolved ts [#1576](#)
 - * Fix the potential deadlock issue for the processor [#2142](#)
 - Backup & Restore (BR)
 - * Fix a bug that all system tables are filtered during restore [#1197](#) [#1201](#)
 - * Fix the issue that Backup & Restore reports the error of “file already exists” when TDE is enabled during the restore [#1179](#)
 - TiDB Lightning
 - * Fix the TiDB Lightning panic issue for some special data [#1213](#)
 - * Fix the EOF error reported when TiDB Lightning splits the imported large CSV files [#1133](#)
 - * Fix a bug that an excessively large base value is generated when TiDB Lightning imports tables with the `auto_increment` column of the `FLOAT` or `DOUBLE` type [#1186](#)
 - * Fix the issue that TiDB fails to parse the `DECIMAL` type data in Parquet files [#1277](#)

14.6.4 TiDB 5.0.2 Release Notes

Release date: June 10, 2021

TiDB version: 5.0.2

14.6.4.1 Compatibility Changes

- Tools
 - TiCDC
 - * Deprecate `--sort-dir` in the `cdc cli changefeed` command. Instead, users can set `--sort-dir` in the `cdc server` command. [#1795](#)

14.6.4.2 New Features

- TiKV
 - Enable the Hibernate Region feature by default [#10266](#)

14.6.4.3 Improvements

- TiDB
 - Avoid frequently reading the `mysql.stats_histograms` table if the cached statistics is up-to-date to avoid high CPU usage [#24317](#)
- TiKV
 - BR now supports the S3-compatible storage using the virtual-host addressing mode [#10243](#)
 - Support the back pressure for TiCDC's scan speed [#10151](#)
 - Reduce the memory usage of TiCDC's initial scan [#10133](#)
 - Improve the cache hit ratio of the TiCDC's Old Value feature in the pessimistic transaction [#10089](#)
 - Split Regions more evenly to mitigate the issue that the growth of Region size exceeds the splitting speed when there are hotspot writes [#9785](#)
- TiFlash
 - Optimize the table lock to prevent DDL jobs and data reads from blocking each other
 - Support casting the `INTEGER` or `REAL` type to `REAL` type
- Tools
 - TiCDC
 - * Add monitoring metrics for the table memory consumption [#1885](#)
 - * Optimize the memory and CPU usages during the sorting stage [#1863](#)
 - * Delete some useless log information that might cause user confusion [#1759](#)
 - Backup & Restore (BR)
 - * Clarify some ambiguous error messages [#1132](#)
 - * Support checking the cluster version of a backup [#1091](#)
 - * Support backing up and restoring system tables in the `mysql` schema [#1143](#) [#1078](#)
 - Dumpling
 - * Fix the issue that no error is output when a backup operation fails [#280](#)

14.6.4.4 Bug Fixes

- TiDB
 - Fix the panic issue caused by using the prefix index and index join in some cases [#24547](#) [#24716](#) [#24717](#)
 - Fix the issue that the prepared plan cache of `point get` is incorrectly used by the `point get` statement in the transaction [#24741](#)
 - Fix the issue of writing the wrong prefix index value when the collation is `ascii_bin` or `latin1_bin` [#24569](#)
 - Fix the issue that the ongoing transaction might be interrupted by the GC worker [#24591](#)
 - Fix a bug that the point query might get wrong on the clustered index when `new-collation` is enabled but `new-row-format` is disabled [#24541](#)
 - Refactor the conversion of partition keys for shuffle hash join [#24490](#)
 - Fix the panic issue that occurs when building the plan for queries that contain the `HAVING` clause [#24045](#)
 - Fix the issue that the column pruning improvement causes the `Apply` and `Join` operators' results to go wrong [#23887](#)
 - Fix a bug that the primary lock fallen back from async commit cannot be resolved [#24384](#)
 - Fix a GC issue of statistics that might cause duplicated fm-sketch records [#24357](#)
 - Avoid unnecessary pessimistic rollback when the pessimistic locking receives the `ErrKeyExists` error [#23799](#)
 - Fix the issue that numeric literals cannot be recognized when the `sql_mode` contains `ANSI_QUOTES` [#24429](#)
 - Forbid statements such as `INSERT INTO table PARTITION (<partitions>)... → ON DUPLICATE KEY UPDATE` to read data from non-listed partitions [#24746](#)
 - Fix the potential `index out of range` error when a SQL statement contains both `GROUP BY` and `UNION` [#24281](#)
 - Fix the issue that the `CONCAT` function incorrectly handles the collation [#24296](#)
 - Fix the issue that the `collation_server` global variable does not take effect in new sessions [#24156](#)
- TiKV
 - Fix a TiCDC OOM issue caused by reading old values [#9996](#) [#9981](#)
 - Fix the issue of empty value in the secondary index for the clustered primary key column when collation is `latin1_bin` [#24548](#)
 - Add the `abort-on-panic` configuration, which allows TiKV to generate the core dump file when panic occurs. Users still need to correctly configure the environment to enable core dump [#10216](#)
 - Fix the performance regression issue of `point get` queries that occurs when TiKV is not busy [#10046](#)
- PD

- Fix the issue that the PD Leader re-election is slow when there are many stores [#3697](#)
- Fix the panic issue that occurs when removing the evict leader scheduler from a non-existent store [#3660](#)
- Fix the issue that the statistics are not updated after offline peers are merged [#3611](#)
- TiFlash
 - Fix the issue of incorrect results when cloning shared delta index concurrently
 - Fix the potential issue that TiFlash fails to restart with incomplete data
 - Fix the issue that old dm files are not removed automatically
 - Fix the potential panic that occurs when the Compaction Filter feature is enabled
 - Fix the potential issue that `ExchangeSender` sends duplicated data
 - Fix the issue that TiFlash cannot resolve the lock fallen back from async commit
 - Fix the issue of incorrect results returned when the casted result of the `TIMEZONE` type contains the `TIMESTAMP` type
 - Fix the TiFlash panic issue that occurs during Segment Split
 - Fix the issue that the execution information about the non-root MPP task is not accurate
- Tools
 - TiCDC
 - * Fix the issue that the time zone information is lost in the Avro output [#1712](#)
 - * Support cleaning up stale temporary files in Unified Sorter and forbid sharing the `sort-dir` directory [#1742](#)
 - * Fix a deadlock bug in the KV client that occurs when many stale Regions exist [#1599](#)
 - * Fix the wrong help information in the `--cert-allowed-cn` flag [#1697](#)
 - * Revert the update for `explicit_defaults_for_timestamp` which requires the `SUPER` privilege when replicating data to MySQL [#1750](#)
 - * Support the sink flow control to reduce the risk of memory overflow [#1840](#)
 - * Fix a bug that the replication task might stop when moving a table [#1828](#)
 - * Fix the issue that the TiKV GC safe point is blocked due to the stagnation of TiCDC changefeed checkpoint [#1759](#)
 - Backup & Restore (BR)
 - * Fix the issue that the `DELETE` events are lost during the log restore [#1063](#)
 - * Fix a bug that causes BR to send too many useless RPC requests to TiKV [#1037](#)
 - * Fix the issue that no error is output when a backup operation fails [#1043](#)
 - TiDB Lightning
 - * Fix the issue of TiDB Lightning panic that occurs when generating KV data [#1127](#)

- * Fix the issue that TiDB Lightning in the TiDB-backend mode cannot load any data when the autocommit is disabled [#1104](#)
- * Fix a bug that the batch split Region fails due to the total key size exceeding the raft entry limit during the data import [#969](#)

14.6.5 TiDB 5.0.1 Release Notes

Release date: April 24, 2021

TiDB version: 5.0.1

14.6.5.1 Compatibility change

- The default value of the **committer-concurrency** configuration item is changed from 16 to 128.

14.6.5.2 Improvements

- TiDB
 - Support the built-in function VITNESS_HASH() [#23915](#)
- TiKV
 - Use zstd to compress the Region snapshot [#10005](#)
- PD
 - Modify the Region score calculator to better satisfy isomeric stores [#3605](#)
 - Avoid unexpected statistics after adding the **scatter region** scheduler [#3602](#)
- Tools
 - Backup & Restore (BR)
 - * Remove some misleading information from the summary log [#1009](#)

14.6.5.3 Bug Fixes

- TiDB
 - Fix the issue that the execution result of project elimination might be wrong when the projection result is empty [#24093](#)
 - Fix the issue of wrong query results when a column contains NULL values in some cases [#24063](#)

- Forbid generating MPP plans when the scan contains virtual columns [#24058](#)
- Fix the wrong reuse of `PointGet` and `TableDual` in Plan Cache [#24043](#)
- Fix the error that occurs when the optimizer builds the `IndexMerge` plan for clustered indexes [#24042](#)
- Fix the type inference of the BIT-type errors [#24027](#)
- Fix the issue that some optimizer hints do not take effect when the `PointGet` operator exists [#23685](#)
- Fix the issue that DDL operations might fail when rolling back due to an error [#24080](#)
- Fix the issue that the index range of the binary literal constant is incorrectly built [#24041](#)
- Fix the potential wrong results of the IN clause in some cases [#24023](#)
- Fix the wrong results of some string functions [#23879](#)
- Users now need both `INSERT` and `DELETE` privileges on a table to perform `REPLACE` operations [#23939](#)
- Fix the performance regression when executing the point query [#24070](#)
- Fix the wrong `TableDual` plans caused by incorrectly comparing binaries and bytes [#23918](#)

- TiKV

- Fix the issue that the coprocessor fails to properly handle the signed or unsigned integer types in the IN expression [#10018](#)
- Fix the issue of many empty Regions after batch ingesting SST files [#10015](#)
- Fix the potential panic that occurs when the input of `cast_string_as_time` is invalid UTF-8 bytes [#9995](#)
- Fix a bug that TiKV cannot start up after the file dictionary file is damaged [#9992](#)

- TiFlash

- Fix the issue that the storage engine fails to remove the data of some ranges
- Fix the issue of incorrect results when casting the time type to the integer type
- Fix a bug that the `receiver` cannot find corresponding tasks within 10 seconds
- Fix the issue that there might be invalid iterators in `cancelMPPQuery`
- Fix a bug that the behavior of the `bitwise` operator is different from that of TiDB
- Fix the alert issue caused by overlapping ranges when using the `prefix key`
- Fix the issue of incorrect results when casting the string type to the integer type
- Fix the issue that consecutive and fast writes might make TiFlash out of memory
- Fix the issue that duplicated column names will make TiFlash raise errors
- Fix the issue that TiFlash fails to parse MPP plans
- Fix the potential issue that the exception of null pointer might be raised during the table GC
- Fix the TiFlash panic issue that occurs when writing data to dropped tables

- Fix the issue that TiFlash might panic during BR restore
- Tools
 - TiDB Lightning
 - * Fix the issue of the inaccurate table count in the progress log during the import [#1005](#)
 - Backup & Restore (BR)
 - * Fix a bug that the actual backup speed exceeds the `--ratelimit` limit [#1026](#)
 - * Fix the issue of backup interruption caused by the failure of a few TiKV nodes [#1019](#)
 - * Fix the issue of the inaccurate table count in the progress log during TiDB Lightning's import [#1005](#)
 - TiCDC
 - * Fix the concurrency issue in Unified Sorter and filter the unhelpful error messages [#1678](#)
 - * Fix a bug that the creation of redundant directories might interrupt the replication with MinIO [#1672](#)
 - * Set the default value of the `explicit_defaults_for_timestamp` session variable to `ON` to make the MySQL 5.7 downstream keep the same behavior with the upstream TiDB [#1659](#)
 - * Fix the issue that the incorrect handling of `io.EOF` might cause replication interruption [#1648](#)
 - * Correct the TiKV CDC endpoint CPU metric in the TiCDC dashboard [#1645](#)
 - * Increase `defaultBufferChanSize` to avoid replication blocking in some cases [#1632](#)

14.6.6 What's New in TiDB 5.0

Release date: April 7, 2021

TiDB version: 5.0.0

In v5.0, PingCAP is dedicated to helping enterprises quickly build applications based on TiDB, freeing them from worries about database performance, performance jitter, security, high availability, disaster recovery, troubleshooting SQL performance, and so on.

In v5.0, the key new features or improvements are as follows:

- Introduce Massively Parallel Processing (MPP) architecture through TiFlash nodes, which shares the execution workloads of large join queries among TiFlash nodes. When the MPP mode is enabled, TiDB, based on cost, determines whether to use the MPP framework to perform the calculation. In the MPP mode, the join keys are redistributed through the `Exchange` operation while being calculated, which distributes the

calculation pressure to each TiFlash node and speeds up the calculation. According to the benchmark, with the same cluster resource, TiDB 5.0 MPP shows 2 to 3 times of speedup over Greenplum 6.15.0 and Apache Spark 3.1.1, and some queries have 8 times better performance.

- Introduce the clustered index feature to improve database performance. For example, in the TPC-C tpmC test, the performance of TiDB, with clustered index enabled, improves by 39%.
- Enable the async commit feature to reduce the write latency. For example, in the 64-thread Sysbench test, the average latency of updating indexes, with async commit enabled, is reduced by 41.7%, from 12.04 ms to 7.01 ms.
- Reduce jitters. This is achieved by improving the optimizer stability and by limiting system tasks' usages of I/O, network, CPU, and memory resources. For example, in the 8-hour performance test, the standard deviation of TPC-C tpmC does not exceed 2%.
- Enhance system stability by improving scheduling and by keeping execution plans stable as much as possible.
- Introduces Raft Joint Consensus algorithm, which ensures the system availability during the Region membership change.
- Optimize EXPLAIN features and invisible index, which helps Database Administrators (DBAs) debug SQL statements more efficiently.
- Guarantee reliability for enterprise data. You can back up data from TiDB to Amazon S3 storage and Google Cloud GCS, or restore data from these cloud storage platforms.
- Improve performance of data import from or data export to Amazon S3 storage or TiDB/MySQL, which helps enterprises quickly build applications on the cloud. For example, in the TPC-C test, the performance of importing 1 TiB data improves by 40%, from 254 GiB/h to 366 GiB/h.

14.6.6.1 Compatibility changes

14.6.6.1.1 System variables

- Add the `tidb_executor_concurrency` system variable to control the concurrency of multiple operators. The previous `tidb_*_concurrency` settings (such as `tidb_projection_concurrency`) still take effect but with a warning when you use them.
- Add the `tidb_skip_ascii_check` system variable to specify whether to skip the ASCII validation check when the ASCII character set is written. This default value is OFF.
- Add the `tidb_enable_strict_double_type_check` system variable to determine whether the syntax like `double(N)` can be defined in the table schema. This default value is OFF.
- Change the default value of `tidb_dml_batch_size` from 20000 to 0. This means that batch DML statements are no longer used by default in `LOAD/INSERT INTO SELECT → ...`. Instead, large transactions are used to comply with the strict ACID semantics.

Note:

The scope of the variable is changed from session to global, and the default value is changed from 20000 to 0. If the application relies on the original default value, you need to use the `set global` statement to modify the variable to the original value after the upgrade.

- Control temporary tables' syntax compatibility using the `tidb_enable_noop_functions` system variable. When this variable value is OFF, the `CREATE TEMPORARY TABLE` syntax returns an error.
- Add the following system variables to directly control the garbage collection-related parameters:
 - `tidb_gc_concurrency`
 - `tidb_gc_enable`
 - `tidb_gc_life_time`
 - `tidb_gc_run_interval`
 - `tidb_gc_scan_lock_mode`
- Change the default value of `enable-joint-consensus` from `false` to `true`, which enables the Joint Consensus feature by default.
- Change the value of `tidb_enable_amend_pessimistic_txn` from 0 or 1 to ON or OFF.
- Change the default value of `tidb_enable_clustered_index` from OFF to INT_ONLY with the following new meanings:
 - ON: clustered index is enabled. Adding or deleting non-clustered indexes is supported.
 - OFF: clustered index is disabled. Adding or deleting non-clustered indexes is supported.
 - INT_ONLY: the default value. The behavior is consistent with that before v5.0. You can control whether to enable clustered index for the INT type together with `alter-primary-key = false`.

Note:

The INT_ONLY value of `tidb_enable_clustered_index` in 5.0 GA has the same meaning as the OFF value in 5.0 RC. After upgrading from a 5.0 RC cluster with the OFF setting to 5.0 GA, it will be displayed as INT_ONLY.

14.6.6.1.2 Configuration file parameters

- Add the `index-limit` configuration item for TiDB. Its value defaults to 64 and ranges between [64, 512]. A MySQL table supports 64 indexes at most. If its value exceeds the default setting and more than 64 indexes are created for a table, when the table schema is re-imported into MySQL, an error will be reported.
- Add the `enable-enum-length-limit` configuration item for TiDB to be compatible and consistent with MySQL's ENUM/SET length (ENUM length < 255). The default value is `true`.
- Replace the `pessimistic-txn.enable` configuration item with the `tidb_txn_mode` environment variable.
- Replace the `performance.max-memory` configuration item with `performance.server ↴ -memory-quota`
- Replace the `tikv-client.copr-cache.enable` configuration item with `tikv-client ↴ .copr-cache.capacity-mb`. If the item's value is 0.0, this feature is disabled. If the item's value is greater than 0.0, this feature is enabled. Its default value is 1000.0.
- Replace the `rocksdb.auto-tuned` configuration item with `rocksdb.rate-limiter- ↴ auto-tuned`.
- Delete the `raftstore.sync-log` configuration item. By default, written data is forcibly spilled to the disk. Before v5.0, you can explicitly disable `raftstore.sync- ↴ log`. Since v5.0, the configuration value is forcibly set to `true`.
- Change the default value of the `gc.enable-compaction-filter` configuration item from `false` to `true`.
- Change the default value of the `enable-cross-table-merge` configuration item from `false` to `true`.
- Change the default value of the `rate-limiter-auto-tuned` configuration item from `false` to `true`.

14.6.6.1.3 Others

- Before the upgrade, check the value of the TiDB configuration `feedback-probability ↴ .` If the value is not 0, the “panic in the recoverable goroutine” error will occur after the upgrade, but this error does not affect the upgrade.
- Forbid conversion between VARCHAR type and CHAR type during the column type change to avoid data correctness issues.

14.6.6.2 New features

14.6.6.2.1 SQL

List partitioning (**Experimental**)

[User document](#)

With the list partitioning feature, you can effectively query and maintain tables with a large amount of data.

With this feature enabled, partitions and how data is distributed among partitions are defined according to the `PARTITION BY LIST(expr)PARTITION part_name VALUES IN (...)` expression. The partitioned tables' data set supports at most 1024 distinct integer values. You can define the values using the `PARTITION ... VALUES IN (...)` clause.

To enable list partitioning, set the session variable `tidb_enable_list_partition` to `ON`.

List COLUMNS partitioning (**Experimental**)

User document

List COLUMNS partitioning is a variant of list partitioning. You can use multiple columns as partition keys. Besides the integer data type, you can also use the columns in the string, DATE, and DATETIME data types as partition columns.

To enable List COLUMNS partitioning, set the session variable `tidb_enable_list_partition` to `ON`.

Invisible indexes

User document, #9246

When you tune performance or select optimal indexes, you can set an index to be `Visible` or `Invisible` by using SQL statements. This setting can avoid performing resource-consuming operations, such as `DROP INDEX` and `ADD INDEX`.

To modify the visibility of an index, use the `ALTER INDEX` statement. After the modification, the optimizer decides whether to add this index to the index list based on the index visibility.

EXCEPT and INTERSECT operators

User document, #18031

The `INTERSECT` operator is a set operator, which returns the intersection of the result sets of two or more queries. To some extent, it is an alternative to the `Inner Join` operator.

The `EXCEPT` operator is a set operator, which combines the result sets of two queries and returns elements that are in the first query result but not in the second.

14.6.6.2.2 Transaction

User document, #18005

In the pessimistic transaction mode, if the tables involved in a transaction contain concurrent DDL operations or `SCHEMA VERSION` changes, the system automatically updates the transaction's `SCHEMA VERSION` to the latest to ensure the successful transaction commit, and to avoid that the client receives the `Information schema is changed` error when the transaction is interrupted by DDL operations or `SCHEMA VERSION` changes.

This feature is disabled by default. To enable the feature, modify the value of `tidb_enable_amend_pessimistic_txn` system variable. This feature is introduced in v4.0.7 and has the following issues fixed in v5.0:

- The compatibility issue that occurs when TiDB Binlog executes `Add Column` operations
- The data inconsistency issue that occurs when using the feature together with the unique index
- The data inconsistency issue that occurs when using the feature together with the added index

Currently, this feature still has the following incompatibility issues:

- Transaction's semantics might change when there are concurrent transactions
- Known compatibility issue that occurs when using the feature together with TiDB Binlog
- Incompatibility with `Change Column`

14.6.6.2.3 Character set and collation

- Support the `utf8mb4_unicode_ci` and `utf8_unicode_ci` collations. [User document](#), [#17596](#)
- Support the case-insensitive comparison sort for collations

14.6.6.2.4 Security

[User document](#), [#18566](#)

To meet security compliance requirements (such as *General Data Protection Regulation*, or GDPR), the system supports desensitizing information (such as ID and credit card number) in the output error messages and logs, which can avoid leaking sensitive information.

TiDB supports desensitizing the output log information. To enable this feature, use the following switches:

- The global variable `tidb_redact_log`. Its default value is 0, which means that desensitization is disabled. To enable desensitization for tidb-server logs, set the variable value to 1.
- The configuration item `security.redact-info-log`. Its default value is `false`, which means that desensitization is disabled. To enable desensitization for tikv-server logs, set the variable value to `true`.
- The configuration item `security.redact_info_log`. Its default value is `false`, which means that desensitization is disabled. To enable desensitization for pd-server logs, set the variable value to `true`.
- The configuration item `security.redact_info_log` for tiflash-server and `security.redact_info_log` for tiflash-learner. Their default values are both `false`, which means that desensitization is disabled. To enable desensitization for tiflash-server and tiflash-learner logs, set the values of both variables to `true`.

This feature is introduced in v5.0. To use the feature, enable the system variable and all configuration items above.

14.6.6.3 Performance optimization

14.6.6.3.1 MPP architecture

User document

TiDB introduces the MPP architecture through TiFlash nodes. This architecture allows multiple TiFlash nodes to share the execution workload of large join queries.

When the MPP mode is on, TiDB determines whether to send a query to the MPP engine for computation based on the calculation cost. In the MPP mode, TiDB distributes the computation of table joins to each running TiFlash node by redistributing the join key during data calculation (Exchange operation), and thus accelerates the calculation. Furthermore, with the aggregation computing feature that TiFlash has already supported, TiDB can pushdown the computation of a query to the TiFlash MPP cluster. Then the distributed environment can help accelerate the entire execution process and dramatically increase the speed of analytic queries.

In the TPC-H 100 benchmark test, TiFlash MPP delivers significant processing speed over analytic engines of traditional analytic databases and SQL on Hadoop. With this architecture, you can perform large-scale analytic queries directly on the latest transaction data, with a higher performance than traditional offline analytic solutions. According to the benchmark, with the same cluster resource, TiDB 5.0 MPP shows 2 to 3 times of speedup over Greenplum 6.15.0 and Apache Spark 3.1.1, and some queries have 8 times better performance.

Currently, the main features that the MPP mode does not support are as follows (For details, refer to [Use TiFlash](#)):

- Table partitioning
- Window Function
- Collation
- Some built-in functions
- Reading data from TiKV
- OOM spill
- Union
- Full Outer Join

14.6.6.3.2 Clustered index

User document, #4841

When you are designing table structures or analyzing database behaviors, it is recommended to use the clustered index feature if you find that some columns with primary keys are often grouped and sorted, queries on these columns often return a certain range of data or a small amount of data with different values, and the corresponding data does not cause read or write hotspot issues.

Clustered indexes, also known as *index-organized tables* in some database management systems, is a storage structure associated with the data of a table. When creating a clustered

index, you can specify one or more columns from the table as the keys for the index. TiDB stores these keys in a specific structure, which allows TiDB to quickly and efficiently find the rows associated with the keys, thus improves the performance of querying and writing data.

When the clustered index feature is enabled, the TiDB performance improves significantly (for example in the TPC-C tpmC test, the performance of TiDB, with clustered index enabled, improves by 39%) in the following cases:

- When data is inserted, the clustered index reduces one write of the index data from the network.
- When a query with an equivalent condition only involves the primary key, the clustered index reduces one read of index data from the network.
- When a query with a range condition only involves the primary key, the clustered index reduces multiple reads of index data from the network.
- When a query with an equivalent or range condition involves the primary key prefix, the clustered index reduces multiple reads of index data from the network.

Each table can either use a clustered or non-clustered index to sort and store data. The differences of these two storage structures are as follows:

- When creating a clustered index, you can specify one or more columns in the table as the key value of the index. A clustered index sorts and stores the data of a table according to the key value. Each table can have only one clustered index. If a table has a clustered index, it is called a clustered index table. Otherwise, it is called a non-clustered index table.
- When you create a non-clustered index, the data in the table is stored in an unordered structure. You do not need to explicitly specify the key value of the non-clustered index, because TiDB automatically assigns a unique ROWID to each row of data. During a query, the ROWID is used to locate the corresponding row. Because there are at least two network I/O operations when you query or insert data, the performance is degraded compared with clustered indexes.

When table data is modified, the database system automatically maintains clustered indexes and non-clustered indexes for you.

All primary keys are created as non-clustered indexes by default. You can create a primary key as a clustered index or non-clustered index in either of the following two ways:

- Specify the keyword `CLUSTERED` | `NONCLUSTERED` in the statement when creating a table, then the system creates the table in the specified way. The syntax is as follows:

```
CREATE TABLE `t` (`a` VARCHAR(255), `b` INT, PRIMARY KEY (`a`, `b`)
    → CLUSTERED);
```

Or

```
CREATE TABLE `t` (`a` VARCHAR(255) PRIMARY KEY CLUSTERED, `b` INT);
```

You can execute the statement `SHOW INDEX FROM tbl-name` to query whether a table has a clustered index.

- Configure the system variable `tidb_enable_clustered_index` to control the clustered index feature. Supported values are `ON`, `OFF`, and `INT_ONLY`.
 - `ON`: Indicates that the clustered index feature is enabled for all types of primary keys. Adding and dropping non-clustered indexes are supported.
 - `OFF`: Indicates that the clustered index feature is disabled for all types of primary keys. Adding and dropping non-clustered indexes are supported.
 - `INT_ONLY`: The default value. If the variable is set to `INT_ONLY` and `alter-primary-key` is set to `false`, the primary keys which consist of single integer columns are created as clustered indexes by default. The behavior is consistent with that of TiDB v5.0 and earlier versions.

If a `CREATE TABLE` statement contains the keyword `CLUSTERED` | `NONCLUSTERED`, the statement overrides the configuration of the system variable and the configuration item.

You are recommended to use the clustered index feature by specifying the keyword `CLUSTERED` | `NONCLUSTERED` in statements. In this way, it is more flexible for TiDB to use all data types of clustered and non-clustered indexes in the system at the same time as required.

It is not recommended to use `tidb_enable_clustered_index = INT_ONLY`, because `INT_ONLY` is temporarily used to make this feature compatible and will be deprecated in the future.

Limitations for the clustered index are as follows:

- Mutual conversion between clustered indexes and non-clustered indexes is not supported.
- Dropping clustered indexes is not supported.
- Adding, dropping, and altering clustered indexes using `ALTER TABLE` statements are not supported.
- Reorganizing and re-creating a clustered index is not supported.
- Enabling or disabling indexes is not supported, which means the invisible index feature is not effective for clustered indexes.
- Creating a `UNIQUE KEY` as a clustered index is not supported.
- Using the clustered index feature together with TiDB Binlog is not supported. After TiDB Binlog is enabled, TiDB only supports creating a single integer primary key as a clustered index. TiDB Binlog does not replicate data changes of existing tables with clustered indexes to the downstream.

- Using the clustered index feature together with the attributes `SHARD_ROW_ID_BITS` and `PRE_SPLIT_REGIONS` is not supported.
- If the cluster is upgraded to a later version then rolls back, you need to downgrade newly-added tables by exporting table data before the rollback and importing the data after the rollback. Other tables are not affected.

14.6.6.3.3 Async Commit

[User document, #8316](#)

The client of the database will wait for the database system to complete the transaction commit in two phases (2PC) synchronously. The transaction returns the result to the client after the first phase commit is successful, and the system executes the second phase commit operation in the background asynchronously to reduce the transaction commit latency. If the transaction write involves only one Region, the second phase is omitted directly, and the transaction becomes a one-phase commit.

After the Async Commit feature is enabled, with the same hardware and configuration, when Sysbench is set to test the Update index with 64 threads, the average latency decreases by 41.7% from 12.04ms to 7.01ms.

When Async Commit feature is enabled, to reduce one network interaction latency and improve the performance of data writes, database application developers are recommended to consider reducing the consistency of transactions from linear consistency to [causal consistency](#). The SQL statement to enable causal consistency is `START TRANSACTION WITH ↪ CAUSAL CONSISTENCY`.

After the causal consistency is enabled, with the same hardware and configuration, when Sysbench is set to test `oltp_write_only` with 64 threads, the average latency decreased by 5.6% from 11.86ms to 11.19ms.

After the consistency of transactions is reduced from the linear consistency to causal consistency, if there is no interdependence between multiple transactions in the application, the transactions do not have a globally consistent order.

The Async Commit feature is enabled by default for newly created v5.0 clusters.

This feature is disabled by default for clusters upgraded from earlier versions to v5.0. You can enable this feature by executing the `set global tidb_enable_async_commit = ON;` and `set global tidb_enable_1pc = ON;` statements.

The limitation for the Async Commit feature is as follows:

- Direct downgrade is not supported.

14.6.6.3.4 Enable the Coprocessor cache feature by default

[User document, #18028](#)

In 5.0 GA, the Coprocessor cache feature is enabled by default. After this feature is enabled, to reduce the latency of reading data, TiDB caches the calculation results of the operators pushed down to tikv-server in tidb-server.

To disable the Coprocessor cache feature, you can modify the `capacity-mb` configuration item of `tikv-client.copr-cache` to 0.0.

14.6.6.3.5 Improve the execution performance of `delete from table where id <? Limit ?` statement

[#18028](#)

The p99 performance of the `delete from table where id <? limit ?` statement is improved by 4 times.

14.6.6.3.6 Optimize load base split strategy to solve the performance problem that data cannot be split in some small table hotspot read scenarios

[#18005](#)

14.6.6.4 Improve stability

14.6.6.4.1 Optimize the performance jitter issue caused by imperfect scheduling

[#18005](#)

The TiDB scheduling process occupies resources such as I/O, network, CPU, and memory. If TiDB does not control the scheduled tasks, QPS and delay might cause performance jitter due to resource preemption.

After the following optimizations, in the 8-hour performance test, the standard deviation of TPC-C tpmC does not exceed 2%.

Introduce new scheduling calculation formulas to reduce unnecessary scheduling and performance jitter

When the node capacity is always near the waterline set in the system, or when the `store-limit` is set too large, to balance the capacity load, the system frequently schedules Regions to other nodes or even schedules Regions back to their original nodes. Because scheduling occupies resources, such as I/O, network, CPU, and memory, and causes performance jitter, this type of scheduling is not necessary.

To mitigate this issue, PD introduces a new set of default scheduling calculation formulas. You can switch back to the old formulas by configuring `region-score-formula-version ↴ = v1`.

Enable the cross-table Region merge feature by default

User document

Before v5.0, TiDB disables the cross-table Region merge feature by default. Starting from v5.0, this feature is enabled by default to reduce the number of empty Regions and the overhead of network, memory, and CPU. You can disable this feature by modifying the `schedule.enable-cross-table-merge` configuration item.

Enable the system to automatically adjust the data compaction speed by default to balance the contention for I/O resources between background tasks and foreground reads and writes

[User document](#)

Before v5.0, to balance the contention for I/O resources between background tasks and foreground reads and writes, the feature that the system automatically adjusts the data compaction speed is disabled by default. Starting from v5.0, TiDB enables this feature by default and optimizes the algorithm so that the latency jitter is significantly reduced.

You can disable this feature by modifying the `rate-limiter-auto-tuned` configuration item.

Enable the GC Compaction Filter feature by default to reduce GC's consumption of CPU and I/O resources

[User document, #18009](#)

When TiDB performs garbage collection (GC) and data compaction, partitions occupy CPU and I/O resources. Overlapping data exists during the execution of these two tasks.

To reduce GC's consumption of CPU and I/O resources, the GC Compaction Filter feature combines these two tasks into one and executes them in the same task. This feature is enabled by default. You can disable it by configuring `gc.enable-compaction-filter = ↴ false`.

TiFlash limits the compression and data sorting's use of I/O resources ([experimental feature](#))

This feature alleviates the contention for I/O resources between background tasks and foreground reads and writes.

This feature is disabled by default. You can enable this feature by modifying the `bg_task_io_rate_limit` configuration item.

Improve the performance of checking scheduling constraints and the performance of fixing the unhealthy Regions in a large cluster

14.6.6.4.2 Ensure that the execution plans are unchanged as much as possible to avoid performance jitter

[User document](#)

SQL Binding supports the `INSERT`、`REPLACE`、`UPDATE`、`DELETE` statements

When tuning performance or maintaining the database, if you find that the system performance is unstable due to unstable execution plans, you can select a manually optimized

SQL statement according to your judgement or tested by `EXPLAIN ANALYZE`. You can bind the optimized SQL statement to the SQL statement to be executed in the application code to ensure stable performance.

When manually binding SQL statements using the `SQL BINDING` statement, you need to ensure that the optimized SQL statement has the same syntax as the original SQL statement.

You can view the manually or automatically bound execution plan information by running the `SHOW {GLOBAL | SESSION} BINDINGS` command. The output is the same as that of versions earlier than v5.0.

Automatically capture and bind execution plans

When upgrading TiDB, to avoid performance jitter, you can enable the baseline capturing feature to allow the system to automatically capture and bind the latest execution plan and store it in the system table. After TiDB is upgraded, you can export the bound execution plan by running the `SHOW GLOBAL BINDING` command and decide whether to delete these plans.

This feature is disabled by default. You can enable it by modifying the server or setting the `tidb_capture_plan_baselines` global system variable to `ON`. When this feature is enabled, the system fetches the SQL statements that appear at least twice from the Statement Summary every `bind-info-lease` (the default value is `3s`), and automatically captures and binds these SQL statements.

14.6.6.4.3 Improve stability of TiFlash queries

Add a system variable `tidb_allow_fallback_to_tikv` to fall back queries to TiKV when TiFlash fails. The default value is `OFF`.

14.6.6.4.4 Improve TiCDC stability and alleviate the OOM issue caused by replicating too much incremental data

[User document, #1150](#)

In TiCDC v4.0.9 or earlier versions, replicating too much data change might cause OOM. In v5.0, the Unified Sorter feature is enabled by default to mitigate OOM issues caused by the following scenarios:

- The data replication task in TiCDC is paused for a long time, during which a large amount of incremental data is accumulated and needs to be replicated.
- The data replication task is started from an early timestamp, so it becomes necessary to replicate a large amount of incremental data.

Unified Sorter is integrated with the `memory/file` sort-engine options of earlier versions. You do not need to manually configure the change.

Limitations:

- You need to provide sufficient disk capacity according to the amount of your incremental data. It is recommended to use SSDs with free capacity greater than 128 GB.

14.6.6.5 High availability and disaster recovery

14.6.6.5.1 Improve system availability during Region membership change

[User document](#), #18079, #7587, #2860

In the process of Region membership changes, “adding a member” and “deleting a member” are two operations performed in two steps. If a failure occurs when the membership change finishes, the Regions will become unavailable and an error of foreground application is returned.

The introduced Raft Joint Consensus algorithm can improve the system availability during Region membership change. “adding a member” and “deleting a member” operations during the membership change are combined into one operation and sent to all members. During the change process, Regions are in an intermediate state. If any modified member fails, the system is still available.

This feature is enabled by default. You can disable it by running the `pd-ctl config ↵ set enable-joint-consensus` command to set the `enable-joint-consensus` value to `false`.

14.6.6.5.2 Optimize the memory management module to reduce system OOM risks

Track the memory usage of aggregate functions. This feature is enabled by default. When SQL statements with aggregate functions are executed, if the total memory usage of the current query exceeds the threshold set by `mem-quota-query`, the system automatically performs operations defined by `oom-action`.

14.6.6.5.3 Improve the system availability during network partition

14.6.6.6 Data migration

14.6.6.6.1 Migrate data from S3/Aurora to TiDB

TiDB data migration tools support using Amazon S3 (and other S3-compatible storage services) as the intermediate for data migration and initializing Aurora snapshot data directly into TiDB, providing more options for migrating data from Amazon S3/Aurora to TiDB.

To use this feature, refer to the following documents:

- [Export data to Amazon S3 cloud storage](#), #8
- [Migrate from Amazon Aurora MySQL Using TiDB Lightning](#), #266

14.6.6.6.2 Optimize the data import performance of TiDB Cloud

TiDB Lightning optimizes its data import performance specifically for AWS T1.standard configurations (or equivalent) of TiDB Cloud. Test results show that TiDB Lightning improves its speed of importing 1TB of TPC-C data into TiDB by 40%, from 254 GiB/h to 366 GiB/h.

14.6.6.7 Data sharing and subscription

14.6.6.7.1 Integrate TiDB to Kafka Connect (Confluent Platform) using TiCDC (experimental feature)

[User document](#), #660

To support the business requirements of streaming TiDB data to other systems, this feature enables you to stream TiDB data to the systems such as Kafka, Hadoop, and Oracle.

The Kafka connectors protocol provided by the Confluent platform is widely used in the community, and it supports transferring data to either relational or non-relational databases in different protocols. By integrating TiCDC to Kafka Connect of the Confluent platform, TiDB extends the ability to stream TiDB data to other heterogeneous databases or systems.

14.6.6.8 Diagnostics

[User document](#)

During the troubleshooting of SQL performance issues, detailed diagnostic information is needed to determine the causes of performance issues. Before TiDB 5.0, the information collected by the EXPLAIN statements was not detailed enough. The root causes of the issues can only be determined based on log information, monitoring information, or even on guess, which might be inefficient.

In TiDB v5.0, the following improvements are made to help you troubleshoot performance issues more efficiently:

- Support using the EXPLAIN ANALYZE statement to analyze all DML statements to show the actual performance plans and the execution information of each operator. #18056
- Support using the EXPLAIN FOR CONNECTION statement to check the real-time status of all the SQL statements being executed. For example, you can use the statement to check the execution duration of each operator and the number of processed rows. #18233
- Provide more details about the operator execution in the output of the EXPLAIN ↗ ANALYZE statement, including the number of RPC requests sent by operators, the duration of resolving lock conflicts, network latency, the scanned volume of deleted data in RocksDB, and the hit rate of RocksDB caches. #18663
- Support automatically recording the detailed execution information of SQL statements in the slow log. The execution information in the slow log is consistent with the output information of the EXPLAIN ANALYZE statement, which includes the time consumed by

each operator, the number of processed rows, and the number of sent RPC requests.
[#15009](#)

14.6.6.9 Deployment and maintenance

14.6.6.9.1 Optimize the logic of cluster deployment operations, to help DBAs deploy a set of standard TiDB production cluster faster

User Document

In previous TiDB versions, DBAs using TiUP to deploy TiDB clusters find that the environment initialization is complicated, the checksum configuration is excessive, and the cluster topology file is difficult to edit. All of these issues lead to low deployment efficiency for DBAs. In TiDB v5.0, the TiDB deployment efficiency using TiUP is improved for DBAs through the following items:

- TiUP Cluster supports the `check topo.yaml` command to perform a more comprehensive one-click environment check and provide repair recommendations.
- TiUP Cluster supports the `check topo.yaml --apply` command to automatically repair environmental problems found during the environment check.
- TiUP Cluster supports the `template` command to get the cluster topology template file for DBAs to edit and support modifying the global node parameters.
- TiUP supports editing the `remote_config` parameter using the `edit-config` command to configure remote Prometheus.
- TiUP supports editing the `external_alertmanagers` parameter to configure different AlertManagers using the `edit-config` command.
- When editing the topology file using the `edit-config` subcommand in `tiup-cluster`, you can modify the data types of the configuration item values.

14.6.6.9.2 Improve upgrade stability

Before TiUP v1.4.0, during the upgrade of a TiDB cluster using `tiup-cluster`, the SQL responses of the cluster jitter for a long period of time, and during PD online rolling upgrades, the QPS of the cluster jitter between 10s to 30s.

TiUP v1.4.0 adjusts the logic and makes the following optimizations:

- During the upgrade of PD nodes, TiUP automatically checks the status of the restarted PD node, and then rolls to upgrade the next PD node after confirming that the status is ready.
- TiUP identifies the PD role automatically, first upgrades the PD nodes of the follower role, and finally upgrades the PD Leader node.

14.6.6.9.3 Optimize the upgrade time

Before TiUP v1.4.0, when DBAs upgrade TiDB clusters using `tiup-cluster`, for clusters with a large number of nodes, the total upgrade time is long and cannot meet the upgrade time window requirement for certain users.

Starting from v1.4.0, TiUP optimizes the following items:

- Supports fast offline upgrades using the `tiup cluster upgrade --offline` subcommand.
- Speeds up the Region Leader relocation for users using rolling upgrades during upgrades by default, so that reduces the time of rolling TiKV upgrades.
- Checks the status of the Region monitor using the `check` subcommand before running a rolling upgrade. Ensure that the cluster is in a normal state before the upgrade, thus reducing the probability of upgrade failures.

14.6.6.9.4 Support the breakpoint feature

Before TiUP v1.4.0, when DBAs upgrade TiDB clusters using `tiup-cluster`, if the execution of a command is interrupted, all the upgrade operations have to be performed again from the beginning.

TiUP v1.4.0 supports retrying failed operations from breakpoints using the `tiup-cluster replay` subcommand, to avoid re-executing all operations after an upgrade interruption.

14.6.6.9.5 Enhance the functionalities of maintenance and operations

TiUP v1.4.0 further enhances the functionalities for operating and maintaining TiDB clusters.

- Supports the upgrade or patch operation on the downtime TiDB and DM clusters to adapt to more usage scenarios.
- Adds the `--version` parameter to the `display` subcommand of `tiup-cluster` to get the cluster version.
- When only Prometheus is included in the node being scaled out, the operation of updating the monitoring configuration is not performed, to avoid scale-out failure due to the absence of the Prometheus node.
- Adds user input to the error message when the results of the input TiUP commands are incorrect, so that you can locate the cause of the problem more quickly.

14.6.6.10 Telemetry

TiDB adds cluster usage metrics in telemetry, such as the number of data tables, the number of queries, and whether new features are enabled.

To learn more about details and how to disable this behavior, refer to [telemetry](#).

14.6.7 TiDB 5.0 RC Release Notes

Release date: January 12, 2021

TiDB version: 5.0.0-rc

TiDB v5.0.0-rc is the predecessor version of TiDB v5.0. In v5.0, PingCAP will be dedicated to helping enterprises quickly build applications based on TiDB, freeing them from worries about database performance, performance jitter, security, high availability, disaster recovery, troubleshooting SQL performance, and so on.

In v5.0, the key new features or improvements are as follows:

- Clustered index. When this feature is enabled, database performance is improved. For example, in the TPC-C tpmC test, TiDB's performance, with clustered index enabled, improves by 39%.
- Async commit. When this feature is enabled, the write latency is reduced. For example, in the Sysbench olpt-insert test, the write latency of TiDB, with async commit enabled, is reduced by 37.3%.
- Reduced jitters. This is achieved by improving the optimizer stability and by limiting system tasks' usages of I/O, network, CPU, and memory resources. For example, in the 72-hour performance test, the standard deviation of Sysbench TPS jitter is reduced from 11.09% to 3.36%.
- Raft Joint Consensus algorithm, which ensures the system availability during the Region membership change.
- Optimized EXPLAIN features and invisible index, which helps Database Administrators (DBAs) debug SQL statements more efficiently.
- Guaranteed reliability for enterprise data. You can back up data from TiDB to AWS S3 storage and Google Cloud GCS, or restore data from these cloud storage platforms.
- Improved performance of data import from or data export to AWS S3 storage or TiDB/MySQL, which helps enterprises quickly build applications on the cloud. For example, in the TPC-C test, the performance of importing 1 TiB data improves by 40%, from 254 GiB/h to 366 GiB/h.

14.6.7.1 SQL

14.6.7.1.1 Support clustered index (experimental)

When the clustered index feature is enabled, TiDB performance improves significantly (for example in the TPC-C tpmC test, TiDB's performance, with clustered index enabled, improves by 39%) in the following cases:

- When data is inserted, the clustered index reduces one write of the index data from the network.
- When a query with an equivalent condition only involves the primary key, the clustered index reduces one read of index data from the network.

- When a query with a range condition only involves the primary key, the clustered index reduces multiple reads of index data from the network.
- When a query with an equivalent or range condition involves the primary key prefix, the clustered index reduces multiple reads of index data from the network.

Clustered index defines the physical storage order of data in a table. The data in the table is sorted only according to the definition of the clustered index. Each table has only one clustered index.

Users can enable the clustered index feature by modifying the `tidb_enable_clustered_index` variable. When enabled, the feature takes effect only on newly created tables and applies to the primary key that has multiple columns or is non-integer types in a single column. If the primary key is an integer type in a single column, or if the table has no primary key, the data is sorted in the same way as before, without being affected by the clustered index.

For example, to check whether a table (`tbl_name`) has a clustered index, execute `select tidb_pk_type from information_schema.tables where table_name = '{tbl_name}'`.

- [User document](#)
- Related issue: [#4841](#)

14.6.7.1.2 Support invisible indexes

When users tune performance or select optimal indexes, they can set an index to be **Visible** or **Invisible** by using SQL statements. This setting can avoid performing resource-consuming operations, such as `DROP INDEX` and `ADD INDEX`.

To modify the visibility of an index, use the `ALTER INDEX` statement. After the modification, the optimizer decides whether to add this index to the index list based on the index visibility.

- [User document](#)
- Related issue: [#9246](#)

14.6.7.1.3 Support EXCEPT and INTERSECT operators

The `INTERSECT` operator is a set operator, which returns the intersection of the result sets of two or more queries. To some extent, it is an alternative to the `InnerJoin` operator.

The `EXCEPT` operator is a set operator, which combines the result sets of two queries and returns elements that are in the first query result but not in the second.

- [User document](#)
- Related issue: [#18031](#)

14.6.7.2 Transaction

14.6.7.2.1 Increase the success rate of executing pessimistic transactions

In the pessimistic transaction mode, if the tables involved in a transaction contain concurrent DDL operations or SCHEMA VERSION changes, the system automatically updates the transaction's SCHEMA VERSION to the latest to avoid the transaction being interrupted by DDL operations and to ensure the successful transaction commit. If the transaction is interrupted, the client receives the Information schema is changed error message.

- [User document](#)
- Related issue: [#18005](#)

14.6.7.3 Character set and collation

Support case-insensitive comparison sort for character sets.

- [User document](#)
- Related issue: [#17596](#)

14.6.7.4 Security

14.6.7.4.1 Support desensitizing error messages and log files

TiDB now supports desensitizing error messages and log files to avoid leaking sensitive information such as ID information and credit card number.

Users can enable the desensitization feature for different components:

- For the TiDB side, set the `tidb_redact_log=1` variable using SQL statements in tidb-server.
- For the TiKV side, set the `security.redact-info-log = true` configuration in tikv-server.
- For the PD side, set the `security.redact-info-log = true` configuration in pd-server. [#2852](#) [#3011](#)
- For the TiFlash side, set the `security.redact_info_log = true` configuration in tiflash-server and set `security.redact-info-log = true` in tiflash-learner.

[User document](#)

Related issue: [#18566](#)

14.6.7.5 Performance improvements

14.6.7.5.1 Support async commit (experimental)

Enabling the async commit feature can significantly reduce the latency of transactions. For example, with this feature enabled, the latency of transactions in the Sysbench oltp-insert test is 37.3% lower than that when this feature is disabled.

Previously without the async commit feature, the statements being written were only returned to the client after the two-phase transaction commit finished. Now the async commit feature supports returning the result to the client after the first phase of the two-phase commit finishes. The second phase is then performed asynchronously in the background, thus reducing the latency of transaction commit.

However, when async commit is enabled, the external consistency of transactions can be guaranteed **only** when `tidb_guarantee_external_consistency = ON` is set. With async commit enabled, the performance might drop.

Users can enable this feature by setting the global variable `tidb_enable_async_commit ↴ = ON`.

- [User document](#)
- Related issue: [#8316](#)

14.6.7.5.2 Improve the optimizer's stability in index selection (experimental)

The optimizer's ability to always select a relatively suitable index greatly determines whether the latency of queries is stable. We have improved and refactored the statistics module to ensure that, for the same SQL statements, the optimizer does not select a different index from multiple candidate indexes each time due to missing or inaccurate statistics. The main improvements to help the optimizer select a relatively suitable index are as follows:

- Add more information to the statistics module, such as the multi-column NDV, the multi-column order dependency, and the multi-column function dependency.
- Refactor the statistics module.
 - Delete TopN values from CMSKetch.
 - Refactor the search logic of TopN.
 - Delete the TopN information from the histogram and create an index of the histogram for easy maintenance of Bucket NDV.

Related issue: [#18065](#)

14.6.7.5.3 Optimize performance jitter caused by imperfect scheduling or imperfect I/O flow control

The TiDB scheduling process occupies resources such as I/O, network, CPU, and memory. If TiDB does not control the scheduled tasks, QPS and delay might cause performance jitter due to resource preemption. After the following optimizations, in the 72-hour test, the standard deviation of Sysbench TPS jitter is reduced from 11.09% to 3.36%.

- Reduce the redundant scheduling issues caused by fluctuations of node capacity (always near the waterline) and caused by PD's `store-limit` configuration value set too large. This is achieved by introducing a new set of scheduling calculation formulas enabled via the `region-score-formula-version = v2` configuration item. [#3269](#)
- Enable the cross-Region merge feature by modifying `enable-cross-table-merge = ↴ true` to reduce the number of empty Regions. [#3129](#)
- Data compaction in the TiKV background occupies a lot of I/O resources. The system automatically adjusts the compaction rate to balance the contention for I/O resources between background tasks and foreground reads and writes. After enabling this feature via the `rate-limiter-auto-tuned` configuration item, the delay jitter is greatly reduced. [#18011](#)
- When TiKV performs garbage collection (GC) and data compaction, partitions occupy CPU and I/O resources. Overlapping data exists during the execution of these two tasks. To reduce I/O usage, the GC Compaction Filter feature combines these two tasks into one and executes them in the same task. This feature is still experimental and you can enable it via `gc.enable-compaction-filter = true`. [#18009](#)
- When TiFlash compresses or sorts data, it occupies a lot of I/O resources. The system alleviates contention for resources by limiting the compression and data sorting's use of I/O resources. This feature is still experimental and you can enable it via `bg_task_io_rate_limit`.

Related issue: [#18005](#)

14.6.7.5.4 Improve the stability of TiFlash in Real-time BI / Data Warehousing scenarios

- Limit the memory usage of DeltaIndex to avoid system out of memory (OOM) caused by excessive memory usage in the scenarios of huge data volume.
- Limit the I/O write traffic used by the background data sorting task to reduce the impact on the foreground tasks.
- Add new thread pools to queue coprocessor tasks, which avoids system OOM caused by excessive memory usage when processing coprocessors in high concurrency.

14.6.7.5.5 Other performance optimizations

- Improve the execution performance of `delete from table where id <?` statement. Its P99 performance improves by four times. [#18028](#)
- TiFlash supports concurrently reading and writing data in multiple local disks to improve performance.

14.6.7.6 High availability and disaster recovery

14.6.7.6.1 Improve system availability during Region membership change (experimental)

In the process of Region membership changes, “adding a member” and “deleting a member” are two operations performed in two steps. If a failure occurs when the membership change finishes, the Regions will become unavailable and an error of foreground application is returned. The introduced Raft Joint Consensus algorithm can improve the system availability during Region membership change. “adding a member” and “deleting a member” operations during the membership change are combined into one operation and sent to all members. During the change process, Regions are in an intermediate state. If any modified member fails, the system is still available. Users can enable this feature by modifying the membership variable by executing `pd-ctl config set enable-joint-consensus ↵ true`. #7587 #2860

- [User document](#)
- Related issue: [#18079](#)

14.6.7.6.2 Optimize the memory management module to reduce system OOM risks

- Reduce the memory consumption of caching statistics.
- Reduce the memory consumption of exporting data using the Dumpling tool.
- Reduced the memory consumption by storing the encrypted intermediate results of data to the disk.

14.6.7.7 Backup and restore

- The Backup & Restore tool (BR) supports backing up data to AWS S3 and Google Cloud GCS. ([User document](#))
- The Backup & Restore tool (BR) supports restoring data from AWS S3 and Google Cloud GCS to TiDB. ([User document](#))
- Related issue: [#89](#)

14.6.7.8 Data import and export

- TiDB Lightning supports importing Aurora snapshot data from AWS S3 storage to TiDB. (Related issue: [#266](#))
- In the TPC-C test of importing 1 TiB of data into DBaaS T1.standard, the performance improves by 40%, from 254 GiB/h to 366 GiB/h.
- Dumpling supports exporting data from TiDB/MySQL to AWS S3 storage (experimental) (Related issue: [#8](#), [User document](#))

14.6.7.9 Diagnostics

14.6.7.9.1 Optimized EXPLAIN features with more collected information help users troubleshoot performance issues

When users troubleshoot SQL performance issues, they need detailed diagnostic information to determine the causes of performance issues. In previous TiDB versions, the information collected by the EXPLAIN statements was not detailed enough. DBAs performed troubleshooting only based on log information, monitoring information, or even on guess, which might be inefficient. The following improvements are made in TiDB v5.0 to help users troubleshoot performance issues more efficiently:

- EXPLAIN ANALYZE supports analyzing all DML statements and shows the actual performance plans and the execution information of each operator. [#18056](#)
- Users can use EXPLAIN FOR CONNECTION to analyze the status information of the SQL statements that are being executed. This information includes the execution duration of each operator and the number of processed rows. [#18233](#)
- More information is available in the output of EXPLAIN ANALYZE, including the number of RPC requests sent by operators, the duration of resolving lock conflicts, network latency, the scanned volume of deleted data in RocksDB, and the hit rate of RocksDB caches. [#18663](#)
- The detailed execution information of SQL statements is recorded in the slow log, which is consistent with the output information of EXPLAIN ANALYZE. This information includes the time consumed by each operator, the number of processed rows, and the number of sent RPC requests. [#15009](#)

User document

14.6.7.10 Deployment and maintenance

- Previously, when the configuration information of TiDB Ansible was imported to TiUP, TiUP put the user configuration in the `ansible-imported-configs` directory. When users later needed to edit the configuration using `tiup cluster edit-config`, the imported configuration was not displayed in the editor interface, which could be confusing for users. In TiDB v5.0, when TiDB Ansible configuration is imported, TiUP puts the configuration information both in `ansible-imported-configs` and in the editor interface. With this improvement, users can see the imported configuration when they are editing the cluster configuration.
- Enhanced `mirror` command that supports merging multiple mirrors into one, publishing components in the local mirror, and adding component owners in the local mirror. [#814](#)
 - For a large enterprise, especially for the financial industry, any change in the production environment is given careful consideration. It can be troublesome if each version requires users to use a CD for installation. In TiDB v5.0, the `merge →` command of TiUP supports merging multiple installation packages into one, which makes the installation easier.

- In v4.0, users had to start `tiup-server` to publish the self-built mirror, which was not convenient enough. In v5.0, users can publish the self-built mirror simply by using `tiup mirror set` to set the current mirror to the local mirror.

14.7 v4.0

14.7.1 TiDB 4.0.16 Release Notes

Release date: December 17, 2021

TiDB version: 4.0.16

14.7.1.1 Compatibility changes

- TiKV
 - Before v4.0.16, when TiDB converts an illegal UTF-8 string to a Real type, an error is reported directly. Starting from v4.0.16, TiDB processes the conversion according to the legal UTF-8 prefix in the string [#11466](#)
- Tools
 - TiCDC
 - * Change the default value of Kafka Sink `max-message-bytes` to 1 MB to prevent TiCDC from sending too large messages to Kafka clusters [#2962](#)
 - * Change the default value of Kafka Sink `partition-num` to 3 so that TiCDC distributes messages across Kafka partitions more evenly [#3337](#)

14.7.1.2 Improvements

- TiDB
 - Upgrade the Grafana version from 7.5.7 to 7.5.11
- TiKV
 - Reduce disk space consumption by adopting the zstd algorithm to compress SST files when restoring data using Backup & Restore or importing data using Local-backend of TiDB Lightning [#11469](#)
- Tools
 - Backup & Restore (BR)
 - * Improve the robustness of restoring [#27421](#)

- TiCDC
 - * Add a tick frequency limit to EtcdWorker to prevent frequent etcd writes from affecting PD services [#3112](#)
 - * Optimize rate limiting control on TiKV reloads to reduce gRPC congestion during changefeed initialization [#3110](#)

14.7.1.3 Bug fixes

- TiDB
 - Fix the query panic caused by overflow in the statistics module when converting a range to points for cost estimation [#23625](#)
 - Fix wrong results of the control functions (such as IF and CASE WHEN) when using the ENUM type data as parameters of such functions [#23114](#)
 - Fix the issue that the GREATEST function returns inconsistent results due to different values of `tidb_enable_vectorized_expression` (on or off) [#29434](#)
 - Fix the panic when applying index join on prefix indexes in some cases [#24547](#)
 - Fix the issue that planner might cache invalid plans for `join` in some cases [#28087](#)
 - Fix a bug that TiDB cannot insert null into a non-null column when `sql_mode` is empty [#11648](#)
 - Fix the wrong result type of the GREATEST and LEAST functions [#29019](#)
 - Fix the privilege check fail error when performing the grant and revoke operations to grant and revoke global level privileges [#29675](#)
 - Fix the panic when using the CASE WHEN function on the ENUM data type [#29357](#)
 - Fix wrong results of the microsecond function in vectorized expressions [#29244](#)
 - Fix wrong results of the hour function in vectorized expression [#28643](#)
 - Fix the issue that optimistic transaction conflicts might cause transactions to block each other [#11148](#)
 - Fix the issue of incomplete log information from the auto analyze result [#29188](#)
 - Fix the issue that using an invalid default date does not report an error when the SQL_MODE is ‘NO_ZERO_IN_DATE’ [#26766](#)
 - Fix the issue that the Coprocessor Cache panel in Grafana does not display metrics. Now, Grafana displays the number of hits/miss/evict [#26338](#)
 - Fix the issue that concurrently truncating the same partition causes DDL statements to stuck [#26229](#)
 - Fix the issue that the length information is wrong when converting Decimal to String [#29417](#)
 - Fix the issue of an extra column in the query result when NATURAL JOIN is used to join multiple tables [#29481](#)
 - Fix the issue that TopN is wrongly pushed down to `indexPlan` when `IndexScan` uses a prefix index [#29711](#)
 - Fix the issue that retrying transactions with the auto-increment columns of DOUBLE type causes data corruption [#29892](#)
- TiKV

- Fix a panic issue that occurs when Region merge, ConfChange, and Snapshot happen at the same time in extreme conditions [#11475](#)
 - Fix the issue of negative sign when the decimal divide result is zero [#29586](#)
 - Fix the issue that the average latency of the by-instance gRPC requests is inaccurate in TiKV metrics [#11299](#)
 - Fix the issue of TiCDC panic that occurs when the downstream database is missing [#11123](#)
 - Fix the issue that the Raft connection is broken when the channel is full [#11047](#)
 - Fix the issue that TiDB cannot correctly identify whether the Int64 types in Max/Min functions are a signed integer or not, which causes the wrong calculation result of Max/Min [#10158](#)
 - Fix the issue that CDC adds scan retries frequently due to the Congest error [#11082](#)
- PD
 - Fix a panic issue that occurs after the TiKV node is removed [#4344](#)
 - Fix slow leader election caused by stucked region syncer [#3936](#)
 - Support that the evict leader scheduler can schedule regions with unhealthy peers [#4093](#)
 - TiFlash
 - Fix the issue that TiFlash fails to start up on some platforms due to the absence of library ns1
 - Tools
 - TiDB Binlog
 - * Fix the bug that Drainer exits when transporting a transaction greater than 1 GB [#28659](#)
 - TiCDC
 - * Fix the negative value error in the changefeed checkpoint lag [#3010](#)
 - * Fix OOM in container environments [#1798](#)
 - * Fix the TiCDC replication interruption issue when multiple TiKVs crash or during a forced restart [#3288](#)
 - * Fix the memory leak issue after processing DDLs [#3174](#)
 - * Fix the issue that changefeed does not fail fast enough when the ErrGCT-TLExceeded error occurs [#3111](#)
 - * Fix the issue that TiCDC replication task might terminate when the upstream TiDB instance unexpectedly exits [#3061](#)
 - * Fix the issue that TiCDC process might panic when TiKV sends duplicate requests to the same Region [#2386](#)
 - * Fix the issue that the volume of Kafka messages generated by TiCDC is not constrained by max-message-size [#2962](#)

- * Fix the issue that `tikv_cdc_min_resolved_ts_no_change_for_1m` keeps alerting when there is no changefeed [#11017](#)
- * Fix the issue that TiCDC sync task might pause when an error occurs during writing a Kafka message [#2978](#)
- * Fix the issue that some partitioned tables without valid indexes might be ignored when `force-replicate` is enabled [#2834](#)
- * Fix the memory leak issue when creating a new changefeed [#2389](#)
- * Fix the issue that might cause inconsistent data due to Sink components advancing resolved ts early [#3503](#)
- * Fix the issue that scanning stock data might fail due to TiKV performing GC when scanning stock data takes too long [#2470](#)
- * Fix the issue that the changefeed update command does not recognize global command line parameters [#2803](#)

14.7.2 TiDB 4.0.15 Release Notes

Release Date: September 27, 2021

TiDB version: 4.0.15

14.7.2.1 Compatibility changes

- TiDB
 - Fix the issue that executing `SHOW VARIABLES` in a new session is slow. This fix reverts some changes made in [#21045](#) and might cause compatibility issues. [#24326](#)
 - The following bug fixes change execution results, which might cause upgrade incompatibilities:
 - * Fix the issue that `greatest(datetime)union null` returns empty string [#26532](#)
 - * Fix the issue that the `having` clause might not work correctly [#26496](#)
 - * Fix the wrong execution results that occur when the collations around the `between` expression are different [#27146](#)
 - * Fix the result wrong that occurs when the argument of the `extract` function is a negative duration [#27236](#)
 - * Fix the wrong execution results that occur when the column in the `group_concat` function has a non-bin collation [#27429](#)
 - * Fix the issue that column information is missed when converting the `Apply` operator to `Join` [#27233](#)
 - * Fix the issue of unexpected behavior when casting the invalid string to DATE [#26762](#)
 - * Fix a bug that the `count distinct` result on multiple columns is wrong when the new collation is enabled [#27091](#)

14.7.2.2 Feature enhancement

- TiKV
 - Support changing TiCDC configurations dynamically [#10645](#)

14.7.2.3 Improvements

- TiDB
 - Trigger auto-analyze based on the histogram row count [#24237](#)
- TiKV
 - Handle read ready and write ready separately to reduce read latency [#10475](#)
 - The slow log of TiKV coprocessor only considers the time spent on processing requests. [#10841](#)
 - Drop log instead of blocking threads when the logger thread is overloaded and the queue is filled up [#10841](#)
 - Reduce the size of Resolved TS messages to save network bandwidth [#2448](#)
- PD
 - Improve the performance of synchronizing Region information between PDs [#3932](#)
- Tools
 - Backup & Restore (BR)
 - * Split and scatter Regions concurrently to improve restore speed [#1363](#)
 - * Retry BR tasks when encountering the PD request error or the TiKV I/O timeout error [#27787](#)
 - * Reduce empty Regions when restoring many small tables to avoid affecting cluster operations after the restore [#1374](#)
 - * Perform the `rebase auto id` operation while creating tables, which saves the separate `rebase auto id` DDL operation and speeds up restore [#1424](#)
 - Dumpling
 - * Filter the skipped databases before getting the table information to improve the filtering efficiency of `SHOW TABLE STATUS` [#337](#)
 - * Use `SHOW FULL TABLES` to get table information for tables to be exported, because `SHOW TABLE STATUS` cannot work properly in some MySQL versions [#322](#)

- * Support backing up MySQL-compatible databases that do not support the START TRANSACTION ... WITH CONSISTENT SNAPSHOT or the SHOW CREATE → TABLE syntax [#309](#)
- * Refine the Dumpling warning log to avoid the misleading information that a dump fails [#340](#)
- TiDB Lightning
 - * Support importing data into tables that have expression index or the index that depends on virtual generated columns [#1404](#)
- TiCDC
 - * Always pulls old values from TiKV internally to improve usability [#2397](#)
 - * Reduce the goroutine usage when a table's Regions are all transferred away from a TiKV node [#2284](#)
 - * Optimize workerpool for fewer goroutines when concurrency is high [#2211](#)
 - * Execute DDL statements asynchronously to avoid affecting other changefeeds [#2295](#)
 - * Add a global gRPC connection pool and share gRPC connections among KV clients [#2531](#)
 - * Fail fast for unrecoverable DML errors [#1724](#)
 - * Optimize memory management when the Unified Sorter is using memory to sort data [#2553](#)
 - * Add Prometheus metrics for DDL executions [#2595](#) [#2669](#)
 - * Prohibit operating TiCDC clusters across major or minor versions [#2601](#)
 - * Remove `file sorter` [#2325](#)
 - * Clean up changefeed metrics when a changefeed is removed, and clean up processor metrics when a processor exits [#2156](#)
 - * Optimize the lock-resolving algorithm after a Region is initialized [#2188](#)

14.7.2.4 Bug fixes

- TiDB
 - Fix a bug that collation is incorrectly set for binary literals when building ranges [#23672](#)
 - Fix the “index out of range” error that occurs when a query includes both GROUP → BY and UNION [#26553](#)
 - Fix the issue that TiDB might fail to send requests if TiKV has tombstone stores [#23676](#) [#24648](#)
 - Remove the undocumented /debug/sub-optimal-plan HTTP API [#27264](#)
 - Fix the issue of wrong character set and collation for the case when expression [#26662](#)
- TiKV

- Fix the issue that BR reports the “file already exists” error when TDE is enabled during data restore [#1179](#)
- Fix the potential disk full issue caused by corrupted snapshot files [#10813](#)
- Fix the issue that TiKV deletes stale Regions too frequently [#10680](#)
- Fix the issue that TiKV frequently reconnects the PD client [#9690](#)
- Check stale file information from the encryption file dictionary [#9115](#)
- PD
 - Fix the issue that PD does not fix the down peers in time [#4077](#)
 - Fix a bug that PD might panic when scaling out TiKV [#3868](#)
- TiFlash
 - Fix the potential issue of data inconsistency that occurs when TiFlash is deployed on multiple disks
 - Fix a bug of incorrect results that occurs when queries contain filters like `CONSTANT ↪ , <, <=, >, >=, or COLUMN`
 - Fix the issue that the store size in metrics is inaccurate under heavy writing
 - Fix a potential bug that TiFlash cannot restore data when deployed on multiple disks
 - Fix the potential issue that TiFlash cannot garbage-collect the delta data after running for a long time
- Tools
 - Backup & Restore (BR)
 - * Fix a bug that the average speed is inaccurately calculated for backup and restore [#1405](#)
 - TiCDC
 - * Fix the `ErrSchemaStorageTableMiss` error that occurs when the DDL Job duplication is encountered in the integrated test [#2422](#)
 - * Fix a bug that a changefeed cannot be removed if the `ErrGCTTLExceeded` error occurs [#2391](#)
 - * Fix the issue that outdated capture might appear in the output of the `capture ↪ list` command [#2388](#)
 - * Fix the deadlock issue in the TiCDC processor [#2017](#)
 - * Fix a data inconsistency issue that occurs because multiple processors might write data to the same table when this table is being re-scheduled [#2230](#)
 - * Fix a bug that the `EtcdbWorker` snapshot isolation is violated in metadata management [#2557](#)
 - * Fix the issue that the changefeed cannot be stopped due to the DDL sink error [#2552](#)
 - * Fix the issue of TiCDC Open Protocol: TiCDC outputs an empty value when there is no change in a transaction [#2612](#)

- * Fix a bug that causes TiCDC to panic on the unsigned TINYINT type [#2648](#)
- * Decrease the gRPC window size to avoid the OOM that occurs when TiCDC captures too many Regions [#2202](#)
- * Fix the OOM issue that occurs when TiCDC captures too many Regions [#2673](#)
- * Fix the issue of process panic that occurs when encoding the data types such as `mysql.TypeString`, `mysql.TypeVarString`, `mysql.TypeVarchar` into JSON [#2758](#)
- * Fix the a memory leak issue that might occur when creating a new changefeed [#2389](#)
- * Fix a bug that DDL handling fails when a changefeed starts at the finish TS of a schema change [#2603](#)
- * Fix the issue of potential DDL loss when the owner crashes when executing DDL statements [#1260](#)
- * Fix the issue of insecure concurrent access to the map in `SinkManager` [#2298](#)

14.7.3 TiDB 4.0.14 Release Notes

Release date: July 27, 2021

TiDB version: 4.0.14

14.7.3.1 Compatibility changes

- TiDB
 - Change the default value of `tidb_multi_statement_mode` from `WARN` to `OFF` in v4.0. It is recommended to use the multi-statement feature of your client library instead. See [the documentation on `tidb_multi_statement_mode`](#) for details. [#25749](#)
 - Upgrade Grafana dashboard from v6.1.16 to v7.5.7 to solve two security vulnerabilities. See the [Grafana blog post](#) for details.
 - Change the default value of the `tidb_stmt_summary_max_stmt_count` variable from 200 to 3000 [#25872](#)
- TiKV
 - Change the default value of `merge-check-tick-interval` from 10 to 2 to speed up the Region merge process [#9676](#)

14.7.3.2 Feature enhancements

- TiKV

- Add a metric `pending` to monitor the number of pending PD heartbeats, which helps locate the issue of slow PD threads [#10008](#)
 - Support using the virtual-host addressing mode to make BR support the S3-compatible storage [#10242](#)
- TiDB Dashboard
 - Support OIDC SSO. By setting the OIDC-compatible SSO services (such as Okta and Auth0), users can log into TiDB Dashboard without entering the SQL password. [#960](#)
 - Add the **Debug API** UI, which is an alternative method to the command line to call several common TiDB and PD internal APIs for advanced debugging [#927](#)

14.7.3.3 Improvements

- TiDB
 - Change the `LOCK` record into the `PUT` record for the index keys using `point get` or `batch point get` for UPDATE reads [#26223](#)
 - Support the MySQL system variable `init_connect` and its associated features [#26031](#)
 - Support the stable result mode to make the query results more stable [#26003](#)
 - Support pushing down the built-in function `json_unquote()` to TiKV [#25721](#)
 - Make the SQL Plan Management (SPM) not affected by the character set [#23295](#)
- TiKV
 - Shutdown the status server first to make sure that the client can correctly check the shutdown status [#10504](#)
 - Always respond to stale peers to make sure that these peers are cleared quicker [#10400](#)
 - Limit the TiCDC sink's memory consumption [#10147](#)
 - When a Region is too large, use the even split to speed up the split process [#10275](#)
- PD
 - Reduce the conflicts among multiple schedulers that run at the same time [#3858](#) [#3854](#)
- TiDB Dashboard
 - Update TiDB Dashboard to v2021.07.17.1 [#3882](#)
 - Support sharing the current session as a read-only session to avoid further modification to it [#960](#)
- Tools

- Backup & Restore (BR)
 - * Speed up restore by merging small backup files [#655](#)
- Dumpling
 - * Always split tables using `_tidb_rowid` when the upstream is a TiDB v3.x cluster, which helps reduce TiDB's memory usage [#306](#)
- TiCDC
 - * Improve the error message returned when a PD endpoint misses the certificate [#1973](#)
 - * Make the sorter I/O errors more user-friendly [#1976](#)
 - * Add a concurrency limit on the Region incremental scan in the KV client to reduce the pressure of TiKV [#1926](#)
 - * Add metrics for the table memory consumption [#1884](#)
 - * Add `capture-session-ttl` to the TiCDC server configuration [#2169](#)

14.7.3.4 Bug fixes

- TiDB
 - Fix the issue that the `SELECT` result is incompatible with MySQL when joining a subquery with a `WHERE` clause evaluated to `false` [#24865](#)
 - Fix the calculation error of the `ifnull` function that occurs when the argument is the `ENUM` or `SET` type [#24944](#)
 - Fix the wrong aggregate pruning in some cases [#25202](#)
 - Fix the incorrect result of the merge join operation that might occur when the column is the `SET` type [#25669](#)
 - Fix the issue that TiDB returns wrong results for cartesian join [#25591](#)
 - Fix the panic issue that occurs when `SELECT ... FOR UPDATE` works on a join operation and the join uses a partitioned table [#20028](#)
 - Fix the issue that the cached `prepared` plan is incorrectly used for `point get` [#24741](#)
 - Fix the issue that the `LOAD DATA` statement can abnormally import non-utf8 data [#25979](#)
 - Fix a potential memory leak issue that occurs when accessing the statistics via an HTTP API [#24650](#)
 - Fix a security issue that occurs when executing the `ALTER USER` statement [#25225](#)
 - Fix a bug that the `TIKV_REGION_PEERS` table cannot correctly handle the `DOWN` status [#24879](#)
 - Fix the issue that invalid strings are not truncated when parsing `DateTime` [#22231](#)
 - Fix the issue that the `select into outfile` statement might have no result when the column type is `YEAR` [#22159](#)
 - Fix the issue that the query result might be wrong when `NULL` is in the `UNION` subquery [#26532](#)

- Fix the issue that the projection operator in execution might cause panic in some cases [#26534](#)
- TiKV
 - Fix the issue that the duration calculation might panic on certain platforms [#related-issue](#)
 - Fix the wrong function that casts `DOUBLE` to `DOUBLE` [#25200](#)
 - Fix the issue that the panic log might be lost when using the async logger [#8998](#)
 - Fix the panic issue that occurs when building a snapshot twice if encryption is enabled [#9786](#) [#10407](#)
 - Fix the wrong arguments type of the `json_unquote()` function in the coprocessor [#10176](#)
 - Fix the issues of suspicious warnings during shutdown and the non-deterministic response from Raftstore [#10353](#) [#10307](#)
 - Fix the issue of backup threads leak [#10287](#)
 - Fix the issue that Region split might panic and corrupt the metadata if the split process is too slow and Region merge is on-going [#8456](#) [#8783](#)
 - Fix the issue that the Region heartbeats prevent TiKV from splitting large Regions in some situations [#10111](#)
 - Fix the wrong statistics caused by the format inconsistency of CM Sketch between TiKV and TiDB [#25638](#)
 - Fix the wrong statistics of the `apply wait duration` metric [#9893](#)
 - Fix the “Missing Blob” error after using `delete_files_in_range` in Titan [#10232](#)
- PD
 - Fix a bug that the scheduler might reappear after executing the delete operation [#2572](#)
 - Fix the data race issue that might occur when the scheduler is started before the temporary configuration is loaded [#3771](#)
 - Fix a PD panic issue that might occur during the Region scattering operation [#3761](#)
 - Fix the issue that the priority of some operators is not set correctly [#3703](#)
 - Fix a PD panic issue that might occur when deleting the `evict-leader` scheduler from a non-existent store [#3660](#)
 - Fix the issue that the PD Leader re-election is slow when there are many stores [#3697](#)
- TiDB Dashboard
 - Fix the issue that the **Profiling** UI cannot profile all TiDB instances [#944](#)
 - Fix the issue that the **Statements** UI does not display “Plan Count” [#939](#)
 - Fix the issue that the **Slow Query** UI might display the “unknown field” error after cluster upgrade [#902](#)

- TiFlash
 - Fix the potential panic issue that occurs when compiling DAG requests
 - Fix the panic issue that occurs when the read load is heavy
 - Fix the issue that TiFlash keeps restarting because of the split failure in column storage
 - Fix a potential bug that TiFlash cannot delete the delta data
 - Fix the incorrect results that occur when cloning the shared delta index concurrently
 - Fix a bug that TiFlash fails to restart in the case of incomplete data
 - Fix the issue that the old dm files cannot be removed automatically
 - Fix the panic issue that occurs when executing the `SUBSTRING` function with specific arguments
 - Fix the issue of incorrect results when casting the `INTEGER` type to the `TIME` type

- Tools
 - Backup & Restore (BR)
 - * Fix the issue that the data restore from the `mysql` schema might fail [#1142](#)
 - TiDB Lightning
 - * Fix the issue that TiDB Lightning fails to parse the `DECIMAL` type data in Parquet files [#1276](#)
 - * Fix the EOF error reported when TiDB Lightning splits the imported large CSV files [#1133](#)
 - * Fix a bug that an excessively large base value is generated when TiDB Lightning imports tables with the `auto_increment` column of the `FLOAT` or `DOUBLE` type [#1185](#)
 - * Fix the issue of TiDB Lightning panic that occurs when generating KV data larger than 4 GB [#1128](#)
 - Dumpling
 - * When using Dumpling to export data to the S3 storage, the `s3>ListBucket` permission is no longer required on the entire bucket. The permission is required only on the data source prefix. [#898](#)
 - TiCDC
 - * Fix the issue of extra partition dispatching after adding new table partitions [#2205](#)
 - * Fix the panic issue that occurs when TiCDC fails to read `/proc/meminfo` [#2023](#)
 - * Reduce TiCDC's runtime memory consumption [#2011 #1957](#)
 - * Fix a bug that some MySQL connection might leak after MySQL sink meets the error and pauses [#1945](#)
 - * Fix the issue that TiCDC changefeed cannot be created when start TS is less than current TS minus GC TTL [#1839](#)

- * Reduce memory `malloc` in sort heap to avoid too much CPU overhead [#1853](#)
- * Fix a bug that the replication task might stop when moving a table [#1827](#)

14.7.4 TiDB 4.0.13 Release Notes

Release date: May 28, 2021

TiDB version: 4.0.13

14.7.4.1 New Features

- TiDB
 - Support changing an `AUTO_INCREMENT` column to an `AUTO_RANDOM` one [#24608](#)
 - Add the `infoschema.client_errors_summary` tables to help users keep track of the errors that have been returned to clients [#23267](#)

14.7.4.2 Improvements

- TiDB
 - Avoid frequently reading the `mysql.stats_histograms` table if the cached statistics is up-to-date to avoid high CPU usage [#24352](#)
- TiKV
 - Make the calculation process of `store used size` more precise [#9904](#)
 - Set more Regions in the `EpochNotMatch` message to reduce Region misses [#9731](#)
 - Speed up freeing the memory accumulated in the long-running cluster [#10035](#)
- PD
 - Optimize the metrics of TSO processing time to help users determine whether the TSO processing time at the PD side is too long [#3524](#)
 - Update the dashboard version to v2021.03.12.1 [#3469](#)
- TiFlash
 - Automatically clean archived data to free up disk space
- Tools
 - Backup & Restore (BR)
 - * Support backing up user tables created in the `mysql` schema [#1077](#)
 - * Update `checkVersion` to check the cluster data and the backup data [#1090](#)

- * Tolerate a small number of TiKV node failures during backup [#1062](#)
- TiCDC
 - * Implement the processor flow control to avoid memory overflow (OOM) [#1751](#)
 - * Support cleaning up stale temporary files in Unified Sorter and prevent multiple `cdc server` instances from sharing the same `sort-dir` directory [#1741](#)
 - * Add the HTTP handler for the failpoint [#1732](#)

14.7.4.3 Bug Fixes

- TiDB
 - Fix the panic issue that occurs when the UPDATE statement with a subquery updates the generated column [#24658](#)
 - Fix the issue that causes duplicate query results when using the multi-column index for data reads [#24634](#)
 - Fix the issue that causes wrong query result when using the BIT type constant as the divisor in the DIV expression [#24266](#)
 - Fix the issue that the NO_ZERO_IN_DATE SQL mode does not take effect for the default column value set in DDL statements [#24185](#)
 - Fix an issue which causes wrong query results when using UNION between a BIT type column and an INTEGER type column [#24026](#)
 - Fix the issue that the TableDual plans are mistakenly created when comparing the BINARY type and the CHAR type [#23917](#)
 - Fix the issue that the insert ignore on duplicate statement might unexpectedly delete table records [#23825](#)
 - Fix the issue that the Audit plugin causes TiDB panic [#23819](#)
 - Fix the issue that the HashJoin operator incorrectly processes the collation [#23812](#)
 - Fix the issue of disconnection that occurs when batch_point_get incorrectly handles abnormal values in the pessimistic transaction [#23778](#)
 - Fix the issue of inconsistent indexes that occurs when the tidb_row_format_version → configuration value is set to 1 and the enable_new_collation value is set to true [#23772](#)
 - Fix a bug that occurs when comparing the INTEGER type column with the STRING constant value [#23705](#)
 - Fix the error that occurs when the BIT type column is passed into the approx_percent function [#23702](#)
 - Fix a bug that causes TiDB to mistakenly report the TiKV server timeout error when executing TiFlash batch requests [#23700](#)
 - Fix the issue that the IndexJoin operator returns wrong results on the prefix column index [#23691](#)
 - Fix the issue which causes wrong query results because the collation on the BINARY type column is not properly handled [#23598](#)

- Fix the issue of query panic that occurs when the UPDATE statement contains the join query with the HAVING clause [#23575](#)
- Fix the issue that causes TiFlash to return wrong results when using the NULL constant in the comparison expression [#23474](#)
- Fix the issue of wrong results when comparing the YEAR type column with the STRING constant [#23335](#)
- Fix the issue that group_concat panics when session.group_concat_max_len is set too small [#23257](#)
- Fix the issue of wrong query results that occurs when using the BETWEEN expression for the TIME type column [#23233](#)
- Fix the issue of privilege check in the DELETE statements [#23215](#)
- Fix the issue that no error is reported when inserting invalid strings to the DECIMAL type column [#23196](#)
- Fix the issue of parsing error occurred when inserting data to the DECIMAL type columns [#23152](#)
- Fix the issue that the USE_INDEX_MERGE hint does not take effect [#22924](#)
- Fix a bug that the query returns wrong results when using ENUM or SET columns in the WHERE clause as an filter [#22814](#)
- Fix a bug that the query returns wrong results when using the clustered index and the new collation at the same time [#21408](#)
- Fix the panic that occurs when executing ANALYZE with enable_new_collation enabled [#21299](#)
- Fix the issue that SQL views does not correctly handle the default roles associated with the SQL DEFINER [#24531](#)
- Fix the issue that cancelling DDL jobs gets stuck [#24445](#)
- Fix the issue that the concat function incorrectly handles the collation [#24300](#)
- Fix a bug that the query returns wrong results when the SELECT field has an IN subquery and the subquery's outer side contains NULL tuples [#24022](#)
- Fix a bug that TiFlash is chosen wrongly by the optimizer when TableScan is in descending order [#23974](#)
- Fix a bug that the point_get plan returns the column name that is inconsistent with that of MySQL [#23970](#)
- Fix the issue that executing the show table status statement on a database with a upper-cased name returns wrong results [#23958](#)
- Fix a bug that the users who do not have the INSERT and DELETE privileges on a table at the same time can perform the REPLACE operation [#23938](#)
- Fix the issue that the results of the concat/make_set/insert expressions are wrong because the collation is incorrectly handled [#23878](#)
- Fix the panic that occurs when executing a query on the table that has RANGE partitions [#23689](#)
- Fix the issue: In the cluster of an earlier version, if the tidb_enable_table_partition ↳ variable is set to false, the tables that contain partitions are handled as non-partitioned tables. Executing batch point get queries on this table, when the cluster is upgraded to a later version, causes connection panic. [#23682](#)
- Fix the issue that when TiDB is configured to listen on TCP and UNIX sock-

ets, the remote hosts over the TCP connection are not correctly validated for connection [#23513](#)

- Fix a bug that the non-default collation causes wrong query results [#22923](#)
- Fix a bug that the **Coprocessor Cache** panel of Grafana does not work [#22617](#)
- Fix the error that occurs when the optimizer accesses the statistic cache [#22565](#)

- TiKV

- Fix a bug that TiKV cannot start if the `file_dict` file is not fully written into the disk that has been full [#9963](#)
- Limit TiCDC's scan speed at 128MB/s by default [#9983](#)
- Reduce the memory usage of TiCDC's initial scan [#10133](#)
- Support the back pressure for TiCDC's scan speed [#10142](#)
- Fix a potential OOM issue by avoiding unnecessary reads to get TiCDC old values [#10031](#)
- Fix a TiCDC OOM issue caused by reading old values [#10197](#)
- Add a timeout mechanism for S3 storages to avoid the client hanging without responses [#10132](#)

- TiFlash

- Fix the issue that number of `delta-merge-tasks` is not reported to Prometheus
- Fix the TiFlash panic issue that occurs during `Segment Split`
- Fix the issue that the `Region write Duration (write blocks)` panel in Grafana is shown in a wrong place
- Fix the potential issue that the storage engine fails to remove data
- Fix the issue of incorrect results when casting the `TIME` type to the `INTEGER` type
- Fix a bug that the behavior of the `bitwise` operator is different from that of TiDB
- Fix the issue of incorrect results when casting the `STRING` type to the `INTEGER` type
- Fix the issue that consecutive and fast writes might make TiFlash out of memory
- Fix the potential issue that the exception of null pointer might be raised during the table GC
- Fix the TiFlash panic issue that occurs when writing data to dropped tables
- Fix the TiFlash panic issue that occurs during BR restore
- Fix a bug that the weights of some characters are wrong when using the general CI collation
- Fix the potential issue that data will be lost in tombstoned tables
- Fix the issue of incorrect results when comparing the string which contains zero bytes
- Fix the issue that the logical function returns wrong results if the input column contains null constants
- Fix the issue that the logical function only accepts the numeric type
- Fix the issue of incorrect results that occurs when the timestamp value is 1970-01-01 and the timezone offset is negative

- Fix the issue that hash value of Decimal256 is not stable
- Tools
 - TiCDC
 - * Fix the deadlock issue caused by the flow control when the sorter's input channel has been blocked [#1779](#)
 - * Fix the issue that the TiKV GC safe point is blocked due to the stagnation of TiCDC changefeed checkpoint [#1756](#)
 - * Revert the update in `explicit_defaults_for_timestamp` which requires the SUPER privilege when replicating data to MySQL [#1749](#)
 - TiDB Lightning
 - * Fix a bug that TiDB Lightning's TiDB-backend cannot load any data when autocommit is disabled

14.7.5 TiDB 4.0.12 Release Notes

Release date: April 2, 2021

TiDB version: 4.0.12

14.7.5.1 New Features

- TiFlash
 - Add tools to check the status of `tiflash replica` for online rolling updates

14.7.5.2 Improvements

- TiDB
 - Refine the output information of the EXPLAIN statement for the `batch cop` mode [#23164](#)
 - Add the warning information for expressions that cannot be pushed to the storage layer in the output of the EXPLAIN statement [#23020](#)
 - Migrate a part of the DDL package code from `Execute/ExecRestricted` to the safe API (2) [#22935](#)
 - Migrate a part of the DDL package code from `Execute/ExecRestricted` to the safe API (1) [#22929](#)
 - Add `optimization-time` and `wait-TS-time` into the slow log [#22918](#)
 - Support querying `partition_id` from the `infoschema.partitions` table [#22489](#)
 - Add `last_plan_from_binding` to help the users know whether a SQL statement's execution plan is matched with the hints in the binding [#21430](#)

- Scatter truncated tables without the `pre-split` option [#22872](#)
- Add three format specifiers for the `str_to_date` expression [#22812](#)
- Record the PREPARE execution failure as Failed Query OPM in the metrics monitor [#22672](#)
- Do not report errors for the PREPARE execution if `tidb_snapshot` is set [#22641](#)
- TiKV
 - Prevent a large number of reconnections in a short period of time [#9879](#)
 - Optimize the write operations and Batch Get in the scenarios of many tombstones [#9729](#)
 - Change the default value of `leader-transfer-max-log-lag` to 128 to increase the success rate of leader transfer [#9605](#)
- PD
 - Update the Region cache only when `pending-peers` or `down-peers` changes, which reduces the pressure of updating heartbeats [#3471](#)
 - Prevent the Regions in `split-cache` from becoming the target of merge [#3459](#)
- TiFlash
 - Optimize the configuration file and remove useless items
 - Reduce the size of TiFlash binary files
 - Use an adaptive aggressive GC strategy to reduce memory usage
- Tools
 - TiCDC
 - * Add a double confirmation when users create or resume the changefeed with the `start-ts` or `checkpoint-ts` 1 day before the current timestamp [#1497](#)
 - * Add Grafana panels for the Old Value feature [#1571](#)
 - Backup & Restore (BR)
 - * Log the `HTTP_PROXY` and `HTTPS_PROXY` environmental variables [#827](#)
 - * Improve the backup performance when there are many tables [#745](#)
 - * Report errors if the service safe point check fails [#826](#)
 - * Add the `cluster_version` and `br_version` information in `backupmeta` [#803](#)
 - * Add retry for external storage errors to increase the success rate of backup [#851](#)
 - * Reduce memory usage during backup [#886](#)
 - TiDB Lightning
 - * Check the TiDB cluster version before running TiDB Lightning to avoid unexpected errors [#787](#)
 - * Fail fast when TiDB Lightning meets the `cancel` error [#867](#)

- * Add `tikv-importer.engine-mem-cache-size` and `tikv-importer.local- ↳ writer-mem-cache-size` configuration items to balance between memory usage and performance [#866](#)
- * Run `batch split region` in parallel for TiDB Lightning's Local-backend to increase the import speed [#868](#)
- * When using TiDB Lightning to import data from a S3 storage, TiDB Lightning no longer requires the `s3>ListBucket` permission [#919](#)
- * When resuming from a checkpoint, TiDB Lightning keeps using the original engine [#924](#)

14.7.5.3 Bug Fixes

- TiDB
 - Fix the issue that the `get` variable expression goes wrong when the session variable is hexadecimal literals [#23372](#)
 - Fix the issue that wrong collation is used when creating the fast execution plan for the `Enum` or `Set` type [#23292](#)
 - Fix the possible wrong result of the `nullif` expression when it is used with `is- ↳ null` [#23279](#)
 - Fix the issue that the auto-analysis is triggered outside its time range [#23219](#)
 - Fix the issue that the `CAST` function might ignore errors for the `point get` plan [#23211](#)
 - Fix a bug that prevents SPM from taking effect when `CurrentDB` is empty [#23209](#)
 - Fix the issue of possible wrong table filters for the `IndexMerge` plan [#23165](#)
 - Fix the issue of unexpected `NotNullFlag` in the returning types of the `NULL` constant [#23135](#)
 - Fix a bug that collation might not be handled by the `text` type [#23092](#)
 - Fix the issue that the range partition might incorrectly handle the `IN` expression [#23074](#)
 - Fix the issue that after marking a TiKV store as tombstone, starting new TiKV stores with different StoreIDs with the same IP address and port keeps returning the `StoreNotMatch` error [#23071](#)
 - Do not adjust the `INT` type when it is `NULL` and compared with `YEAR` [#22844](#)
 - Fix the issue of lost connection when loading data on tables with the `auto_random` column [#22736](#)
 - Fix the issue of DDL hangover when the DDL operation meets panic in the cancelling path [#23297](#)
 - Fix the wrong key range of index scan when comparing the `YEAR` column with `NULL` [#23104](#)
 - Fix the issue that a successfully created view is failed to use [#23083](#)
- TiKV

- Fix the issue that the `IN` expression does not properly handle unsigned/signed integers [#9850](#)
- Fix the issue that the ingest operation is not re-entrant [#9779](#)
- Fix the issue that the space is missed when converting JSON to string in TiKV coprocessor [#9666](#)
- PD
 - Fix a bug that the isolation level is wrong when the store lacks the label [#3474](#)
- TiFlash
 - Fix the issue of incorrect execution results when the default value of the `binary` type column contains leading or tailing zero bytes
 - Fix a bug that TiFlash fails to synchronize schema if the name of the database contains special characters
 - Fix the issue of incorrect results when handling the `IN` expression with decimal values
 - Fix a bug that the metric for the opened file count shown in Grafana is high
 - Fix a bug that TiFlash does not support the `Timestamp` literal
 - Fix the potential not responding issue while handling the `FROM_UNIXTIME` expression
 - Fix the issue of incorrect results when casting string as integer
 - Fix a bug that the `like` function might return wrong results
- Tools
 - TiCDC
 - * Fix a disorder issue of the `resolved ts` event [#1464](#)
 - * Fix a data loss issue caused by wrong table scheduling due to the network problem [#1508](#)
 - * Fix a bug of untimely release of resources after a processor is stopped [#1547](#)
 - * Fix a bug that the transaction counter is not correctly updated, which might cause database connection leak [#1524](#)
 - * Fix the issue that multiple owners can co-exist when PD has jitter, which might lead to table missing [#1540](#)
 - Backup & Restore (BR)
 - * Fix a bug that `WalkDir` for the s3 storage returns `nil` if the target path is bucket name [#733](#)
 - * Fix a bug that the `status` port is not served with TLS [#839](#)
 - TiDB Lightning
 - * Fix the error that TiKV Importer might ignore that the file has already existed [#848](#)

- * Fix a bug that the TiDB Lightning might use the wrong timestamp and read the wrong data [#850](#)
- * Fix a bug that TiDB Lightning's unexpected exit might cause damaged checkpoint file [#889](#)
- * Fix the issue of possible data error that occurs because the `cancel` error is ignored [#874](#)

14.7.6 TiDB 4.0.11 Release Notes

Release date: February 26, 2021

TiDB version: 4.0.11

14.7.6.1 New Features

- TiDB
 - Support the `utf8_unicode_ci` and `utf8mb4_unicode_ci` collations [#22558](#)
- TiKV
 - Support the `utf8mb4_unicode_ci` collation [#9577](#)
 - Support the `cast_year_as_time` collation [#9299](#)
- TiFlash
 - Add a Coprocessor thread pool to queue Coprocessor requests for execution, which avoids out of memory (OOM) in some cases, and add the `cop_pool_size` ↵ and `batch_cop_pool_size` configuration items with the default values of `NumOfPhysicalCores * 2`

14.7.6.2 Improvements

- TiDB
 - Reorder inner joins that are simplified from outer joins [#22402](#)
 - Support multiple clusters in Grafana dashboards [#22534](#)
 - Add a workaround for the issue of multiple statements [#22468](#)
 - Divide the metrics of slow query into `internal` and `general` [#22405](#)
 - Add interface for `utf8_unicode_ci` and `utf8mb4_unicode_ci` collations [#22099](#)
- TiKV
 - Add metrics of server information for DBaaS [#9591](#)
 - Support multiple clusters in Grafana dashboards [#9572](#)

- Report RocksDB metrics to TiDB [#9316](#)
- Record the suspension time for Coprocessor tasks [#9277](#)
- Add thresholds of key counts and key size for Load Base Split [#9354](#)
- Check whether the file exists before data import [#9544](#)
- Improve Fast Tune panels [#9180](#)
- PD
 - Support multiple clusters in Grafana dashboards [#3398](#)
- TiFlash
 - Optimize the performance of the `date_format` function
 - Optimize the memory consumption of handling ingest SST
 - Optimize the retrying logic in Batch Coprocessor to reduce the probability of Region error
- Tools
 - TiCDC
 - * Add the version information in the `capture` metadata and add the CLI version of a `changefeed` in the `changefeed` metadata [#1342](#)
 - TiDB Lightning
 - * Create tables in parallel to improve import performance [#502](#)
 - * Skip splitting Regions to improve import performance if the engine's total size is smaller than the Region size [#524](#)
 - * Add a import progress bar and optimize the accuracy of restore progress [#506](#)

14.7.6.3 Bug Fixes

- TiDB
 - Fix the issue of abnormal `unicode_ci` constant propagation [#22614](#)
 - Fix the issue that might cause wrong collation and coercibility [#22602](#)
 - Fix the issue that might cause wrong collation results [#22599](#)
 - Fix the issue of constant substitution for different collations [#22582](#)
 - Fix a bug that the `like` function might return wrong result when using collation [#22531](#)
 - Fix the issue of incorrect `duration` type inference in `least` and `greatest` functions [#22580](#)
 - Fix a bug that occurs when the `like` function handles a single character wildcard `(_)` followed by a multiple character wildcard `(%)` [#22575](#)
 - Fix the type inference error of the TiDB's built-in functions (`least` and `greatest`) [#22562](#)

- Fix a bug that makes the `like` function get the wrong result if the pattern string is a unicode string [#22529](#)
- Fix a bug that the point get query does not get the snapshot data when the `@@tidb_snapshot` variable is set [#22527](#)
- Fix the potential panic that occurs when generating hints from joins [#22518](#)
- Fix the issue that strings are incorrectly converted to the `BIT` type [#22420](#)
- Fix the `index out of range` error that occurs when inserting values to the `tidb_rowid` column [#22359](#)
- Fix a bug that the cached plan is incorrectly used [#22353](#)
- Fix the runtime panic in the `WEIGHT_STRING` function when the length of the binary/char string is too large [#22332](#)
- Forbid using the generated column when the number of function parameters is invalid [#22174](#)
- Correctly set the process information before building the execution plan [#22148](#)
- Fix the issue of inaccurate runtime statistics of `IndexLookUp` [#22136](#)
- Add cache for the memory usage information when the cluster is deployed in a container [#22116](#)
- Fix the issue of the decoding plan errors [#22022](#)
- Report errors for using invalid window specifications [#21976](#)
- Report errors when the `PREPARE` statement is nested with `EXECUTE`, `DEALLOCATE` or `PREPARE` [#21972](#)
- Fix the issue that no error is reported when the `INSERT IGNORE` statement is used on a non-existing partition [#21971](#)
- Unify the encoding of `EXPLAIN` results and slow log [#21964](#)
- Fix the issue of unknown columns in join when using the aggregate operator [#21957](#)
- Fix the wrong type inference in the `ceiling` function [#21936](#)
- Fix the issue that the `Double` type column ignores its decimal [#21916](#)
- Fix the issue that the correlated aggregation is calculated in subqueries [#21877](#)
- Report errors for the JSON object with key length ≥ 65536 [#21870](#)
- Fix the issue that the `dynname` function is incompatible with MySQL [#21850](#)
- Fix the issue that the `to_base64` function returns `NULL` when the input data is too long [#21813](#)
- Fix the failure of comparing multiple fields in the subquery [#21808](#)
- Fix the issue that occurs when comparing the float type in JSON [#21785](#)
- Fix the issue that occurs when comparing the types of JSON objects [#21718](#)
- Fix the issue that the coercibility value of the `cast` function is incorrectly set [#21714](#)
- Fix an unexpected panic when using the `IF` function [#21711](#)
- Fix the issue that the `NULL` result returned from JSON search is incompatible with MySQL [#21700](#)
- Fix the issue that occurs when checking the `only_full_group_by` mode using `ORDER BY` and `HAVING` [#21697](#)
- Fix the issue that the units of `Day` and `Time` are incompatible with MySQL [#21676](#)
- Fix the issue that the default values of `LEAD` and `LAG` cannot adapt to the field

type [#21665](#)

- Perform a check to ensure that the LOAD DATA statement can only load data into base tables [#21638](#)
- Fix the issue that occurs when `addtime` and `subtime` functions handle invalid arguments [#21635](#)
- Change the round rule for approximate values to “round to the nearest even number” [#21628](#)
- Fix the issue that `WEEK()` does not recognize `@@GLOBAL.default_week_format` until it has been explicitly read [#21623](#)

- TiKV

- Fix the issue that TiKV is failed to build with `PROST=1` [#9604](#)
- Fix the unmatched memory diagnostics [#9589](#)
- Fix the issue that the end key of a partial RawKV-restore range is inclusive [#9583](#)
- Fix the issue of TiKV panic that occurs when loading the old value of a key of a rolled-back transaction during TiCDC’s incremental scan [#9569](#)
- Fix the configuration glitch of old values when changefeeds with different settings connect to one Region [#9565](#)
- Fix a crash issue that occurs when running a TiKV cluster on a machine with a network interface that lacks the MAC address (introduced in v4.0.9) [#9516](#)
- Fix the issue of TiKV OOM when backing up a huge Region [#9448](#)
- Fix the issue that `region-split-check-diff` cannot be customized [#9530](#)
- Fix the issue of TiKV panic when the system time goes back [#9542](#)

- PD

- Fix the issue that member health metrics are incorrectly displayed [#3368](#)
- Forbid removing the tombstone store that still has peers [#3352](#)
- Fix the issue that the store limit cannot be persisted [#3403](#)
- Fix the limit constriction of the scatter range scheduler [#3401](#)

- TiFlash

- Fix a bug that the `min/max` result is wrong for the decimal type
- Fix a bug that TiFlash might crash when reading data
- Fix the issue that some data written after DDL operations might be lost after data compaction
- Fix the issue that TiFlash incorrectly handles decimal constants in Coprocessor
- Fix the potential crash during the learner read process
- Fix the inconsistent behaviors of division by 0 or NULL between TiDB and TiFlash

- Tools

- TiCDC

- * Fix a bug that the TiCDC service might unexpectedly exit when `ErrTaskStatusNotExists` and the closing of `capture` session occur at the same time [#1240](#)
- * Fix the old value switch issue that a `changefeed` might be affected by another `changefeed` [#1347](#)
- * Fix a bug that the TiCDC service might hang when processing a new `changefeed` with the invalid `sort-engine` parameter [#1309](#)
- * Fix the issue of panic that occurs when getting the debugging information on non-owner nodes [#1349](#)
- * Fix the issue that the `ticdc_processor_num_of_tables` and `ticdc_processor_table_res` → metrics are not properly updated when adding or removing tables [#1351](#)
- * Fix the issue of potential data loss if a processor crashes when adding a table [#1363](#)
- * Fix a bug that the owner might lead to abnormal TiCDC server exits during table migrations [#1352](#)
- * Fix a bug that TiCDC does not exit in time after the service GC safepoint is lost [#1367](#)
- * Fix a bug that the KV client might skip creating the event feed [#1336](#)
- * Fix a bug that the atomicity of transactions is broken when the transactions are replicated to the downstream [#1375](#)
- Backup & Restore (BR)
 - * Fix the issue that TiKV might be caused to generate a big Region after BR restores the backup [#702](#)
 - * Fix the issue that BR restores a table's Auto ID even if the table does not have Auto ID [#720](#)
- TiDB Lightning
 - * Fix a bug that `column count mismatch` might be triggered when using the TiDB-backend [#535](#)
 - * Fix a bug that TiDB-backend panics if the column count of the source file and the column count of the target table mismatch [#528](#)
 - * Fix a bug that TiKV might unexpectedly panic during TiDB Lightning's data import [#554](#)

14.7.7 TiDB 4.0.10 Release Notes

Release date: January 15, 2021

TiDB version: 4.0.10

14.7.7.1 New Features

- PD

- Add the `enable-redact-log` configuration item to redact user data from logs
[#3266](#)
- TiFlash
 - Add the `security.redact_info_log` configuration item to redact user data from logs

14.7.7.2 Improvements

- TiDB
 - Make the size limit of a key-value entry in transaction configurable using `txn-entry-size-limit` [#21843](#)
- PD
 - Optimize the `store-state-filter` metrics to show more information [#3100](#)
 - Upgrade the `go.etcd.io/bbolt` dependency to v1.3.5 [#3331](#)
- Tools
 - TiCDC
 - * Enable the old value feature for the `maxwell` protocol [#1144](#)
 - * Enable the unified sorter feature by default [#1230](#)
 - Dumpling
 - * Support checking unrecognized arguments and printing the current progress during dumping [#228](#)
 - TiDB Lightning
 - * Support retrying the error that occurs when reading from S3 [#533](#)

14.7.7.3 Bug Fixes

- TiDB
 - Fix a concurrency bug that might cause the batch client timeout [#22336](#)
 - Fix the issue of duplicate bindings caused by concurrent baseline capture [#22295](#)
 - Make the baseline capture bound to the SQL statement work when the log level is '`debug`' [#22293](#)
 - Correctly release GC locks when Region merge occurs [#22267](#)
 - Return correct values for user variables of the `datetime` type [#22143](#)
 - Fix the issue of using index merge when there are multiple table filters [#22124](#)

- Fix the wrong precision issue in TiFlash caused by the prepare plan cache [#21960](#)
- Fix the issue of incorrect results caused by schema change [#21596](#)
- Avoid unnecessary column flag changes in `ALTER TABLE` [#21474](#)
- Set the database name for table aliases of query blocks used in optimizer hints [#21380](#)
- Generate the proper optimizer hint for `IndexHashJoin` and `IndexMergeJoin` [#21020](#)
- TiKV
 - Fix the wrong mapping between ready and peer [#9409](#)
 - Fix the issue that some logs are not redacted when `security.redact-info-log` is set to `true` [#9314](#)
- PD
 - Fix the issue that the ID allocation is not monotonic [#3308](#) [#3323](#)
 - Fix the issue that the PD client might be blocked in some cases [#3285](#)
- TiFlash
 - Fix the issue that TiFlash fails to start because TiFlash fails to process the TiDB schema of an old version
 - Fix the issue that TiFlash fails to start due to incorrect handling of `cpu_time` on the RedHat system
 - Fix the issue that TiFlash fails to start when `path_realtime_mode` is set to `true`
 - Fix an issue of incorrect results when calling the `substr` function with three parameters
 - Fix the issue that TiFlash does not support changing the `Enum` type even if the change is lossless
- Tools
 - TiCDC
 - * Fix the `maxwell` protocol issues, including the issue of `base64` data output and the issue of outputting TSO to unix timestamp [#1173](#)
 - * Fix a bug that outdated metadata might cause the newly created changefeed abnormal [#1184](#)
 - * Fix the issue of creating the receiver on the closed notifier [#1199](#)
 - * Fix a bug that the TiCDC owner might consume too much memory in the etcd watch client [#1227](#)
 - * Fix the issue that `max-batch-size` does not take effect [#1253](#)
 - * Fix the issue of cleaning up stale tasks before the capture information is constructed [#1280](#)

- * Fix the issue that the recycling of db conn is block because `rollback` is not called in MySQL sink [#1285](#)
- Dumpling
 - * Avoid TiDB out of memory (OOM) by setting the default behavior of `tidb_mem_quota_query` [#233](#)
- Backup & Restore (BR)
 - * Fix the issue that BR v4.0.9 cannot restore the files backed up using BR v4.0.8 on GCS [#688](#)
 - * Fix the issue that BR panics when the GCS storage URL has no prefix [#673](#)
 - * Disable backup statistics by default to avoid BR OOM [#693](#)
- TiDB Binlog
 - * Fix the issue that when the `AMEND TRANSACTION` feature is enabled, Drainer might choose the incorrect schema version to generate SQL statements [#1033](#)
- TiDB Lightning
 - * Fix a bug that the Region is not split because the Region key is incorrectly encoded [#531](#)
 - * Fix the issue that the failure of `CREATE TABLE` might be lost when multiple tables are created [#530](#)
 - * Fix the issue of `column count mismatch` when using the TiDB-backend [#535](#)

14.7.8 TiDB 4.0.9 Release Notes

Release date: December 21, 2020

TiDB version: 4.0.9

14.7.8.1 Compatibility Changes

- TiDB
 - Deprecate the `enable-streaming` configuration item [#21055](#)
- TiKV
 - Reduce I/O and mutex contention when encryption at rest is enabled. The change is backwardly incompatible. If users need to downgrade the cluster to a version earlier than v4.0.9, `security.encryption.enable-file-dictionary-log` must be disabled and TiKV must be restarted before the downgrade. [#9195](#)

14.7.8.2 New Features

- TiFlash
 - Support storing the latest data of the storage engine on multiple disks (experimental)
- TiDB Dashboard
 - Support displaying and sorting by all fields in the **SQL Statements** page [#749](#)
 - Support zooming and panning the topology graph [#772](#)
 - Support displaying the disk usage information in the **SQL Statements** and **Slow Queries** pages [#777](#)
 - Support exporting list data in the **SQL Statements** and **Slow Queries** pages [#778](#)
 - Support customizing the Prometheus address [#808](#)
 - Add a page for cluster statistics [#815](#)
 - Add more time-related fields in the **Slow Queries** details [#810](#)

14.7.8.3 Improvements

- TiDB
 - Avoid the (index) merge join in a heuristical way when converting equal conditions to other conditions [#21146](#)
 - Differentiate the types of user variables [#21107](#)
 - Support setting the `GOGC` variable in the configuration file [#20922](#)
 - Make the dumped binary time (`Timestamp` and `Datetime`) more compatible with MySQL [#21135](#)
 - Provide an error message for statements that use the `LOCK IN SHARE MODE` syntax [#21005](#)
 - Avoid outputting unnecessary warnings or errors when folding constants in shortcut-able expressions [#21040](#)
 - Raise an error when preparing the `LOAD DATA` statement [#21199](#)
 - Ignore the attribute of the integer zero-fill size when changing the integer column types [#20986](#)
 - Add the executor-related runtime information of DML statements in the result of `EXPLAIN ANALYZE` [#21066](#)
 - Disallow multiple updates on the primary key in a single SQL statements [#21113](#)
 - Add a monitoring metric for the connection idle time [#21301](#)
 - Temporarily enable the slow log when the `runtime/trace` tool is running [#20578](#)
- TiKV
 - Add the tag to trace the source of the `split` command [#8936](#)

- Support dynamically changing the `pessimistic-txn.pipelined` configuration [#9100](#)
- Reduce the impact on performance when running Backup & Restore and TiDB Lightning [#9098](#)
- Add monitoring metrics for the ingesting SST errors [#9096](#)
- Prevent the leader from being hibernated when some peers still need to replicate logs [#9093](#)
- Increase the success rate of the pipelined pessimistic locking [#9086](#)
- Change the default value of `apply-max-batch-size` and `store-max-batch-size` to 1024 [#9020](#)
- Add the `max-background-flushes` configuration item [#8947](#)
- Disable `force-consistency-checks` by default to improve performance [#9029](#)
- Offload the queries on the Region size from `pd heartbeat worker` to split
→ `check worker` [#9185](#)

- PD

- Check the TiKV cluster version when a TiKV stores become `Tombstone`, which prevents users from enabling incompatible features during the process of down-grade or upgrade [#3213](#)
- Disallow the TiKV store of a lower version to change from `Tombstone` back to `Up` [#3206](#)

- TiDB Dashboard

- Keep expanding when “Expand” is clicked for SQL statements [#775](#)
- Open detail pages in new windows for **SQL Statements** and **Slow Queries** [#816](#)
- Improve descriptions for time-related fields in **Slow Queries** details [#817](#)
- Display detailed error messages [#794](#)

- TiFlash

- Reduce the latency of replica reads
- Refine TiFlash’s error messages
- Limit the memory usage of cache data when the data volume is huge
- Add a monitoring metric for the number of coprocessor tasks being handled

- Tools

- Backup & Restore (BR)
 - * Disallow the ambiguous `--checksum false` argument in the command line, which does not correctly disable checksum. Only `--checksum=false` is accepted. [#588](#)
 - * Support changing the PD configuration temporarily so that PD can recover the original configuration after BR accidentally exists [#596](#)

- * Support analyzing tables after restore [#622](#)
- * Retry for the `read index not ready` and `proposal in merging mode errors` [#626](#)
- TiCDC
 - * Add an alert for enabling TiKV’s Hibernate Region feature [#1120](#)
 - * Reduce memory usage in the schema storage [#1127](#)
 - * Add the feature of unified sorter, which accelerates replication when the data size of the incremental scan is large (experimental) [#1122](#)
 - * Support configuring the maximum message size and the maximum message batch in the TiCDC Open Protocol message (only for Kafka sink) [#1079](#)
- Dumpling
 - * Retry dumping data on failed chunks [#182](#)
 - * Support configuring both the `-F` and `-r` arguments at the same time [#177](#)
 - * Exclude system databases in `--filter` by default [#194](#)
 - * Support the `--transactional-consistency` parameter and support rebuilding MySQL connections during retry [#199](#)
 - * Support using the `-c,--compress` parameter to specify the compression algorithm used by Dumpling. An empty string means no compression. [#202](#)
- TiDB Lightning
 - * Filter out all system schemas by default [#459](#)
 - * Support setting a default value for the auto-random primary key for the Local-backend or Importer-backend [#457](#)
 - * Use range properties to make the range split more precise in Local-backend [#422](#)
 - * Support a human-readable format (such as “2.5 GiB”) in `tikv-importer`
 \hookrightarrow `.region-split-size`, `mydumper.read-block-size`, `mydumper.batch-size`, and `mydumper.max-region-size` [#471](#)
- TiDB Binlog
 - * Exit the Drainer process with the non-zero code if the upstream PD is down or if applying DDL or DML statements to the downstream fails [#1012](#)

14.7.8.4 Bug Fixes

- TiDB
 - Fix the issue of incorrect results when using a prefix index with the `OR` condition [#21287](#)
 - Fix a bug that might cause panic when automatic retry is enabled [#21285](#)
 - Fix a bug that occurs when checking partition definition according to column type [#21273](#)
 - Fix a bug that the value type of the partition expression is not consistent with the partition column type [#21136](#)

- Fix a bug that the hash-type partition does not check whether the partition name is unique [#21257](#)
- Fix the wrong results returned after inserting a value of the non-INT type into the hash partitioned table [#21238](#)
- Fix the unexpected error when using index join in the `INSERT` statement in some cases [#21249](#)
- Fix the issue that the `BigInt` unsigned column value in the `CASE WHEN` operator is incorrectly converted to the `BigInt` signed value [#21236](#)
- Fix a bug that index hash join and index merge join do not consider collation [#21219](#)
- Fix a bug that the partitioned table does not consider collation in the `CREATE ↪ TABLE` and `SELECT` syntax [#21181](#)
- Fix the issue that the query result of `slow_query` might miss some rows [#21211](#)
- Fix the issue that `DELETE` might not delete data correctly when the database name is not in a pure lower representation [#21206](#)
- Fix a bug that causes schema change after DML operations [#21050](#)
- Fix the bug that the coalesced column cannot be queried when using join [#21021](#)
- Fix the wrong results of some semi-join queries [#21019](#)
- Fix the issue that the table lock does not take effect on the `UPDATE` statement [#21002](#)
- Fix the issue of stack overflow that occurs when building the recursive view [#21001](#)
- Fix the unexpected result returned when performing index merge join operations on outer join [#20954](#)
- Fix the issue that sometimes a transaction that has an undetermined result might be treated as failed [#20925](#)
- Fix the issue that `EXPLAIN FOR CONNECTION` cannot show the last query plan [#21315](#)
- Fix the issue that when Index Merge is used in a transaction with the Read Committed isolation level, the result might be incorrect [#21253](#)
- Fix the auto-ID allocation failure caused by the transaction retry after the write conflict [#21079](#)
- Fix the issue that JSON data cannot be correctly imported to TiDB using `LOAD ↪ DATA` [#21074](#)
- Fix the issue that the default value of newly added Enum-type columns is incorrect [#20998](#)
- Fix the issue that the `adddate` function inserts invalid characters [#21176](#)
- Fix the issue that the wrong `PointGet` plan generated in some situations causes wrong results [#21244](#)
- Ignore the conversion of daylight saving time in the `ADD_DATE` function to be compatible with MySQL [#20888](#)
- Fix a bug that prevents inserting strings with trailing spaces that exceed `varchar` or `char`'s length constraint [#21282](#)
- Fix a bug that does not converting the integer from [1, 69] to [2001, 2069] or from [70, 99] to [1970, 1999] when comparing `int` with `year` [#21283](#)

- Fix the panic caused by the overflowing result of the `sum()` function when calculating the `Double` type field [#21272](#)
- Fix a bug that `DELETE` fails to add lock on the unique key [#20705](#)
- Fix a bug that snapshot reads hits the lock cache [#21539](#)
- Fix an issue of potential memory leak after reading a lot of data in a long-lived transaction [#21129](#)
- Fix the issue that omitting the table alias in a subquery will have a syntax error returned [#20367](#)
- Fix the issue that when the argument of the `IN` function in a query is the time type, the query might return an incorrect result [#21290](#)
- TiKV
 - Fix the issue that Coprocessor might return wrong results when there are more than 255 columns [#9131](#)
 - Fix the issue that Region Merge might cause data loss during network partition [#9108](#)
 - Fix the issue that the `ANALYZE` statement might cause panic when using the `latin1` character set [#9082](#)
 - Fix the wrong results returned when converting the numeric type to the time type [#9031](#)
 - Fix a bug that TiDB Lightning fails to ingest SST files to TiKV with the Importer-backend or Local-backend when Transparent Data Encryption (TDE) is enabled [#8995](#)
 - Fix the invalid `advertise-status-addr` value (`0.0.0.0`) [#9036](#)
 - Fix the issue that an error is returned indicating that a key exists when this key is locked and deleted in a committed transaction [#8930](#)
 - Fix the issue that the RocksDB cache mapping error causes data corruption [#9029](#)
 - Fix a bug that Follower Read might return stale data after the leader is transferred [#9240](#)
 - Fix the issue that stale old values might be read in the pessimistic lock [#9282](#)
 - Fix a bug that replica read might get stale data after the leader transfer [#9240](#)
 - Fix the issue of TiKV crash that occurs when receiving `SIGPROF` after profiling [#9229](#)
- PD
 - Fix the issue that the leader roles specified using placement rules do not take effect in some cases [#3208](#)
 - Fix the issue that the `trace-region-flow` value is unexpectedly set to `false` [#3120](#)
 - Fix a bug that the service safepoint with infinite Time To Live (TTL) does not work [#3143](#)
- TiDB Dashboard

- Fix a display issue of time in the Chinese language [#755](#)
- Fix a bug that the browser compatibility notice does not work [#776](#)
- Fix the issue that the transaction `start_ts` is incorrectly displayed in some scenarios [#793](#)
- Fix the issue that some SQL texts are incorrectly formatted [#805](#)
- TiFlash
 - Fix the issue that `INFORMATION_SCHEMA.CLUSTER_HARDWARE` might contain the information of disks that are not in use
 - Fix the issue that the estimate on memory usage of Delta Cache is smaller than the actual usage
 - Fix the memory leak caused by thread information statistics
- Tools
 - Backup & Restore (BR)
 - * Fix the failure caused by special characters in S3 secret access keys [#617](#)
 - TiCDC
 - * Fix the issue that multiple owners might exist when the owner campaign key is deleted [#1104](#)
 - * Fix a bug that TiCDC might fail to continue replicating data when a TiKV node crashes or recovers from a crash. This bug only exists in v4.0.8. [#1198](#)
 - * Fix the issue that the metadata is repeatedly flushed to etcd before a table is initialized [#1191](#)
 - * Fix an issue of replication interruption caused by early GC or the latency of updating `TableInfo` when the schema storage caches TiDB tables [#1114](#)
 - * Fix the issue that the schema storage costs too much memory when DDL operations are frequent [#1127](#)
 - * Fix the goroutine leak when a changefeed is paused or stopped [#1075](#)
 - * Increase the maximum retry timeout to 600 seconds in Kafka producer to prevent replication interruption caused by the service or network jitter in the downstream Kafka [#1118](#)
 - * Fix a bug that the Kafka batch size does not take effect [#1112](#)
 - * Fix a bug that some tables' row change might be lost when the network between TiCDC and PD has jitter and when there are paused changefeeds being resumed at the same time [#1213](#)
 - * Fix a bug that the TiCDC process might exit when the network between TiCDC and PD is not stable [#1218](#)
 - * Use a singleton PD client in TiCDC and fix a bug that TiCDC closes PD client by accident which causes replication block [#1217](#)
 - * Fix a bug that the TiCDC owner might consume too much memory in the etcd watch client [#1224](#)
 - Dumpling

- * Fix the issue that Dumpling might get blocked when its connection to the MySQL database server is closed [#190](#)
- TiDB Lightning
 - * Fix the issue that keys are encoded using the wrong field information [#437](#)
 - * Fix the issue that GC life time TTL does not take effect [#448](#)
 - * Fix the issue that causes panic when manually stops the running TiDB Lightning in the Local-backend mode [#484](#)

14.7.9 TiDB 4.0.8 Release Notes

Release date: October 30, 2020

TiDB version: 4.0.8

14.7.9.1 New Features

- TiDB
 - Support the new aggregate function `APPROX_PERCENTILE` [#20197](#)
- TiFlash
 - Support pushing down `CAST` functions
- Tools
 - TiCDC
 - * Support snapshot-level consistent replication [#932](#)

14.7.9.2 Improvements

- TiDB
 - Prioritize low-selectivity indexes in the greedy search procedure of `Selectivity()` [#20154](#)
 - Record more RPC runtime information in Coprocessor runtime statistics [#19264](#)
 - Speed up parsing the slow log to improve query performance [#20556](#)
 - Wait for timeout execution plans during the plan binding stage to record more debug information when the SQL optimizer is verifying potential new plans [#20530](#)
 - Add the execution retry time in the slow log and the slow query result [#20495](#) [#20494](#)
 - Add the `table_storage_stats` system table [#20431](#)
 - Add the RPC runtime statistical information for the `INSERT/UPDATE/REPLACE` statement [#20430](#)

- Add the operator information in the result of `EXPLAIN FOR CONNECTION` [#20384](#)
- Adjust the TiDB error log to the DEBUG level for the client connection/disconnection activities [#20321](#)
- Add monitoring metrics for Coprocessor Cache [#20293](#)
- Add the runtime information of pessimistic lock keys [#20199](#)
- Add two extra sections of time consumption information in the runtime information and `trace` span [#20187](#)
- Add the runtime information of transaction commit in the slow log [#20185](#)
- Disable the index merge join [#20599](#)
- Add the ISO 8601 and timezone supports for temporal string literals [#20670](#)
- TiKV
 - Add the **Fast-Tune** panel page to assist performance diagnostics [#8804](#)
 - Add the `security.redact-info-log` configuration item, which redacts user data from logs [#8746](#)
 - Reformat the metafile of error codes [#8877](#)
 - Enable dynamically changing the `pessimistic-txn.pipelined` configuration [#8853](#)
 - Enable the memory profiling features by default [#8801](#)
- PD
 - Generate the metafile of errors [#3090](#)
 - Add the additional information for the operator [#3009](#)
- TiFlash
 - Add monitoring metrics of Raft logs
 - Add monitoring metrics of memory usage for `cop` tasks
 - Make the `min/max` index more accurate when data is deleted
 - Improve query performance in the case of a small data volume
 - Add the `errors.toml` file to support the standard error code
- Tools
 - Backup and Restore (BR)
 - * Speed up the restore process by pipelining `split` and `ingest` [#427](#)
 - * Support manually restoring PD schedulers [#530](#)
 - * Use `pause` schedulers instead of `remove` schedulers [#551](#)
 - TiCDC
 - * Print statistics in MySQL sink periodically [#1023](#)
 - Dumpling
 - * Support dumpling data directly to S3 storages [#155](#)

- * Support dumping views [#158](#)
- * Support dumping the table that only contains generated columns [#166](#)
- TiDB Lightning
 - * Support multi-byte CSV delimiters and separators [#406](#)
 - * Speed up the restore process by disabling some PD schedulers [#408](#)
 - * Use the GC-TTL API for checksum GC safepoint in the v4.0 cluster to avoid the GC error [#396](#)

14.7.9.3 Bug Fixes

- TiDB
 - Fix the unexpected panic that occurs when using partitioned tables [#20565](#)
 - Fix the wrong result of outer join when filtering the outer side using index merge join [#20427](#)
 - Fix the issue that the NULL value is returned when converting data to the BIT type if the data is too long [#20363](#)
 - Fix the corrupted default value for the BIT type column [#20340](#)
 - Fix the overflow error that might occur when converting the BIT type to the INT64 type [#20312](#)
 - Fix the possible wrong result of the propagate column optimization for the hybrid type column [#20297](#)
 - Fix the panic that might occur when storing outdated plans from the plan cache [#20246](#)
 - Fix the bug that the returned result is mistakenly truncated if FROM_UNIXTIME and UNION ALL are used together [#20240](#)
 - Fix the issue that wrong results might be returned when the Enum type value is converted to the Float type [#20235](#)
 - Fix the possible panic of RegionStore.accessStore [#20210](#)
 - Fix the wrong result returned when sorting the maximum unsigned integer in BatchPointGet [#20205](#)
 - Fix the bug that the coercibilities of Enum and Set are wrong [#20364](#)
 - Fix an issue of ambiguous YEAR conversion [#20292](#)
 - Fix the issue of wrong reported result that occurs when the KV duration panel contains store0 [#20260](#)
 - Fix the issue that the Float type data is mistakenly inserted regardless of the out of range error [#20252](#)
 - Fix the bug that the generated column does not handle bad NULL values [#20216](#)
 - Fix the inaccurate error information for the YEAR type data that is out of range [#20170](#)
 - Fix the unexpected invalid auto-id error that might occur during the pessimistic transaction retry [#20134](#)
 - Fix the issue that the constraint is not checked when using ALTER TABLE to change the Enum/Set type [#20046](#)

- Fix the wrong runtime information of cop tasks recorded when multiple operators are used for concurrency [#19947](#)
 - Fix the issue that read-only system variables cannot be explicitly selected as the session variables [#19944](#)
 - Fix the issue that the duplicate ORDER BY condition might cause sub-optimal execution plans [#20333](#)
 - Fix the issue that the generated metric profile might fail if the font size exceeds the maximum allowable value [#20637](#)
- TiKV
 - Fix the bug that the mutex conflict in encryption causes pd-worker to process heartbeats slowly [#8869](#)
 - Fix the issue that the memory profile is mistakenly generated [#8790](#)
 - Fix the failure to back up databases on GCS when the storage class is specified [#8763](#)
 - Fix the bug that a learner cannot find a leader when the Region is restarted or newly split [#8864](#)
 - PD
 - Fix a bug that Key Visualizer of TiDB Dashboard might cause PD panic in some cases [#3096](#)
 - Fix the bug that PD might panic if a PD store is down for more than 10 minutes [#3069](#)
 - TiFlash
 - Fix the issue of wrong timestamp in the log message
 - Fix the issue that during the multi-disk TiFlash deployment, the wrong capacity causes the creation of TiFlash replicas to fail
 - Fix the bug that TiFlash might throw errors about broken data files after restart
 - Fix the issue that broken files might be left on disk after TiFlash crashes
 - Fix the bug that it might take a long time to wait for index during learner reads if the proxy cannot catch up with the latest Raft lease information
 - Fix the bug that the proxy writes too much Region state information to the key-value engine while replaying the outdated Raft log
 - Tools
 - Backup and Restore (BR)
 - * Fix the `send on closed channel` panic during restore [#559](#)
 - TiCDC
 - * Fix the unexpected exit caused by the failure to update the GC safepoint [#979](#)

- * Fix the issue that the task status is unexpectedly flushed because of the incorrect mod revision cache [#1017](#)
- * Fix the unexpected empty Maxwell messages [#978](#)
- TiDB Lightning
 - * Fix the issue of wrong column information [#420](#)
 - * Fix the infinity loop that occurs when retrying to get Region information in the local mode [#418](#)

14.7.10 TiDB 4.0.7 Release Notes

Release date: September 29, 2020

TiDB version: 4.0.7

14.7.10.1 New Features

- PD
 - Add the `GetAllMembers` function in the PD client to get PD member information [#2980](#)
- TiDB Dashboard
 - Support generating the metrics relationship graph [#760](#)

14.7.10.2 Improvements

- TiDB
 - Add more runtime information for the `join` operator [#20093](#)
 - Add the hit ratio information of coprocessor cache in `EXPLAIN ANALYZE` [#19972](#)
 - Support pushing down the `ROUND` function to TiFlash [#19967](#)
 - Add the default value of `CMSketch` for `ANALYZE` [#19927](#)
 - Refine error message desensitization [#20004](#)
 - Accept connections from clients using connectors from MySQL 8.0 [#19959](#)
- TiKV
 - Support the JSON log format [#8382](#)
- PD
 - Count schedule operators when they are finished rather than added [#2983](#)
 - Set the `make-up-replica` operator to high priority [#2977](#)

- TiFlash
 - Improve the error handling of the Region meta change that occurs during reads
- Tools
 - TiCDC
 - * Support translating more execution-efficient SQL statements in MySQL sink when the old value feature is enabled [#955](#)
 - Backup & Restore (BR)
 - * Add connection retry when the connection is broken during backup [#508](#)
 - TiDB Lightning
 - * Support dynamically updating the log level via the HTTP interface [#393](#)

14.7.10.3 Bug Fixes

- TiDB
 - Fix a vectorization bug from `and/or/COALESCE` caused by shortcut [#20092](#)
 - Fix the issue that plan digests are the same when the cop task stores are of different types [#20076](#)
 - Fix the wrong behavior of the `!= any()` function [#20062](#)
 - Fix the query error that occurs when the `slow-log` file does not exist [#20051](#)
 - Fix the issue that Region requests continue to retry when the context is canceled [#20031](#)
 - Fix the issue that querying the time type of the `cluster_slow_query` table in streaming request might result in an error [#19943](#)
 - Fix the issue that DML statements using `case when` might cause schema change [#20095](#)
 - Fix the issue that the `prev_stmt` information in slow log is not desensitized [#20048](#)
 - Fix the issue that tidb-server does not release the table lock when it exits abnormally [#20020](#)
 - Fix the incorrect error message that occurs when inserting data of the `ENUM` and `SET` type [#19950](#)
 - Fix the wrong behavior of the `IsTrue` function in some situations [#19903](#)
 - Fix the issue that the `CLUSTER_INFO` system table might not work normally after PD is scaled in or out [#20026](#)
 - Avoid unnecessary warnings or errors when folding constants in `control` expressions [#19910](#)
 - Update the method of updating statistics to avoid Out of Memory (OOM) [#20013](#)
- TiKV

- Fix the issue of unavailable Status API when TLS handshake fails [#8649](#)
- Fix the potential undefined behaviors [#7782](#)
- Fix the possible panic caused by generating snapshots when executing `UnsafeDestroyRange` [#8681](#)
- PD
 - Fix the bug that PD might panic if some Regions have no Leader when `balance ↪ -region` is enabled [#2994](#)
 - Fix the statistical deviation of Region size and Region keys after Region merge [#2985](#)
 - Fix the incorrect hotspot statistics [#2991](#)
 - Fix the issue that there is no `nil` check in `redirectSchedulerDelete` [#2974](#)
- TiFlash
 - Fix the wrong result of right outer join
- Tools
 - Backup & Restore (BR)
 - * Fix a bug that causes the TiDB configuration to change after the restore process [#509](#)
 - Dumpling
 - * Fix the issue that Dumpling fails to parse metadata when some variables are `NULL` [#150](#)

14.7.11 TiDB 4.0.6 Release Notes

Release date: September 15, 2020

TiDB version: 4.0.6

14.7.11.1 New Features

- TiFlash
 - Support outer join in TiFlash broadcast join
- TiDB Dashboard
 - Add Query Editor and execution UI (experimental) [#713](#)
 - Support store location topology visualization [#719](#)
 - Add cluster configuration UI (experimental) [#733](#)
 - Support sharing the current session [#741](#)

- Support displaying the number of execution plans in SQL Statement list [#746](#)
- Tools
 - TiCDC (GA since v4.0.6)
 - * Support outputting data in the `maxwell` format [#869](#)

14.7.11.2 Improvements

- TiDB
 - Replace error codes and messages with standard errors [#19888](#)
 - Improve the write performance of partitioned table [#19649](#)
 - Record more RPC runtime information in `Cop Runtime` statistics [#19264](#)
 - Forbid creating tables in `metrics_schema` and `performance_schema` [#19792](#)
 - Support adjusting the concurrency of the union executor [#19886](#)
 - Support out join in broadcast join [#19664](#)
 - Add SQL digest for the process list [#19829](#)
 - Switch to the pessimistic transaction mode for autocommit statement retry [#19796](#)
 - Support the `%r` and `%T` data format in `Str_to_date()` [#19693](#)
 - Enable `SELECT INTO OUTFILE` to require the file privilege [#19577](#)
 - Support the `stddev_pop` function [#19541](#)
 - Add the TiDB-Runtime dashboard [#19396](#)
 - Improve compatibility for the `ALTER TABLE` algorithms [#19364](#)
 - Encode `insert/delete/update` plans in the slow log `plan` field [#19269](#)
- TiKV
 - Reduce QPS drop when `DropTable` or `TruncateTable` is being executed [#8627](#)
 - Support generating metafile of error codes [#8619](#)
 - Add performance statistics for cf scan details [#8618](#)
 - Add the `rocksdb perf` context panel in the Grafana default template [#8467](#)
- PD
 - Update TiDB Dashboard to v2020.09.08.1 [#2928](#)
 - Add more metrics for Region and store heartbeat [#2891](#)
 - Change back to the original way to control the low space threshold [#2875](#)
 - Support standard error codes
 - * [#2918](#) [#2911](#) [#2913](#) [#2915](#) [#2912](#)
 - * [#2907](#) [#2906](#) [#2903](#) [#2806](#) [#2900](#) [#2902](#)
- TiFlash

- Add Grafana panels for data replication (apply `Region snapshots` and `ingest`
↪ `SST files`)
- Add Grafana panels for `write stall`
- Add `dt_segment_force_merge_delta_rows` and `dt_segment_force_merge_delta_deletes`
↪ to adjust the threshold of `write stall`
- Support setting `raftstore.snap-handle-pool-size` to 0 in TiFlash-Proxy to
disable applying Region snapshot by multi-thread to reduce memory consumption
during data replication
- Support CN check on `https_port` and `metrics_port`

- Tools

- TiCDC
 - * Skip resolved lock during puller initialization [#910](#)
 - * Reduce PD write frequency [#937](#)
- Backup & Restore (BR)
 - * Add real time cost in summary log [#486](#)
- Dumpling
 - * Support outputting `INSERT` with column names [#135](#)
 - * Unify the `--filesize` and `--statement-size` definitions with those of my-dumper [#142](#)
- TiDB Lightning
 - * Split and ingest Regions in more precise sizes [#369](#)
- TiDB Binlog
 - * Support setting GC time in `go time` package format [#996](#)

14.7.11.3 Bug Fixes

- TiDB
 - Fix an issue of collecting the `tikv_cop_wait` time in metric profile [#19881](#)
 - Fix the wrong result of `SHOW GRANTS` [#19834](#)
 - Fix the incorrect query result of `!= ALL (subq)` [#19831](#)
 - Fix a bug of converting the `enum` and `set` types [#19778](#)
 - Add a privilege check for `SHOW STATS_META` and `SHOW STATS_BUCKET` [#19760](#)
 - Fix the error of unmatched column lengths caused by `builtinGreatestStringSig`
↪ and `builtinLeastStringSig` [#19758](#)
 - If unnecessary errors or warnings occur, the vectorized control expresions fall back
to their scalar execution [#19749](#)
 - Fix the error of the `Apply` operator when the type of the correlation column is
`Bit` [#19692](#)

- Fix the issue that occurs when the user queries `processlist` and `cluster_log` in MySQL 8.0 client [#19690](#)
- Fix the issue that plans of the same type have different plan digests [#19684](#)
- Forbid changing the column type from `Decimal` to `Int` [#19682](#)
- Fix the issue that `SELECT ... INTO OUTFILE` returns the runtime error [#19672](#)
- Fix the incorrect implementation of `builtinRealIsFalseSig` [#19670](#)
- Fix the issue that the partition expression check misses the parentheses expression [#19614](#)
- Fix a query error when there is an `Apply` operator upon `HashJoin` [#19611](#)
- Fix an incorrect result of vectorization that casts `Real` as `Time` [#19594](#)
- Fix the bug that the `SHOW GRANTS` statement shows grants for non-existent users [#19588](#)
- Fix a query error when there is an `Apply` executor upon `IndexLookupJoin` [#19566](#)
- Fix the wrong results when converting `Apply` to `HashJoin` on a partitioned table [#19546](#)
- Fix incorrect results when there is an `IndexLookUp` executor on the inner side of an `Apply` [#19508](#)
- Fix an unexpected panic when using view [#19491](#)
- Fix the incorrect result of the `anti-semi-join` query [#19477](#)
- Fix the bug that the `TopN` statistics is not deleted when the statistics is dropped [#19465](#)
- Fix a wrong result caused by mistaken usage of batch point get [#19460](#)
- Fix the bug that a column cannot be found in `indexLookupJoin` with a virtual generated column [#19439](#)
- Fix an error that different plans of the `select` and `update` queries compare datum [#19403](#)
- Fix a data race for TiFlash work index in Region cache [#19362](#)
- Fix the bug that the `logarithm` function does not show a warning [#19291](#)
- Fix an unexpected error that occurs when TiDB persists data to disks [#19272](#)
- Support using a single partitioned table on the inner side of index join [#19197](#)
- Fix the wrong hash key value generated for decimal [#19188](#)
- Fix the issue that TiDB returns a `no regions` error when table endKey and Region endKey are the same [#19895](#)
- Fix the unexpected success of alter partition [#19891](#)
- Fix the wrong value of the default maximum packet length allowed for pushed down expressions [#19876](#)
- Fix a wrong behavior for the `Max/Min` functions on the `ENUM/SET` columns [#19869](#)
- Fix the read failure from the `tiflash_segments` and `tiflash_tables` system tables when some TiFlash nodes are offline [#19748](#)
- Fix a wrong result of the `Count(col)` aggregation function [#19628](#)
- Fix a runtime error of the `TRUNCATE` operation [#19445](#)
- Fix the issue that `PREPARE statement FROM @Var` will fail when `Var` contains uppercase characters [#19378](#)
- Fix the bug that schema charset modification in an uppercase schema will cause panic [#19302](#)

- Fix the inconsistency of plans between `information_schema.statements_summary` → and `explain`, when the information contains tikv/tiflash [#19159](#)
- Fix the error in tests that the file does not exist for `select into outfile` [#19725](#)
- Fix the issue that `INFORMATION_SCHEMA.CLUSTER_HARDWARE` does not have raid device information [#19457](#)
- Make the `add index` operation that has a generated column with the `case-when` expression can exit normally when it encounters a parse error [#19395](#)
- Fix the bug that the DDL operation takes too long to retry [#19488](#)
- Make statements like `alter table db.t1 add constraint fk foreign key (→ c2)references t2(c1)` execute without first executing `use db` [#19471](#)
- Change the dispatch error from the `Error` to the `Info` message in the server log file [#19454](#)

- TiKV

- Fix the estimation error for a non-index column when collation is enabled [#8620](#)
- Fix the issue that Green GC might miss locks during the process of Region transfer [#8460](#)
- Fix a panic issue that occurs when TiKV runs very slowly during Raft membership change [#8497](#)
- Fix the deadlock issue that occurs between the PD client thread and other threads when calling PD sync requests [#8612](#)
- Upgrade jemalloc to v5.2.1 to address the issue of memory allocation in huge page [#8463](#)
- Fix the issue that the unified thread pool hangs for long-running queries [#8427](#)

- PD

- Add the `initial-cluster-token` configuration to prevent different clusters from communicating with each other during bootstrap [#2922](#)
- Fix the unit of store limit rate when the mode is `auto` [#2826](#)
- Fix the issue that some schedulers persist configuration without solving errors [#2818](#)
- Fix the empty HTTP response in scheduler [#2871 #2874](#)

- TiFlash

- Fix the issue that after renaming the primary key column in previous versions, TiFlash might not start after upgrading to v4.0.4/v4.0.5
- Fix the exceptions that occur after modifying the column's `nullable` attribute
- Fix the crash caused by computing a table's replication status
- Fix the issue that TiFlash is not available for data reads after users applied unsupported DDL operations
- Fix the exceptions caused by unsupported collations which are treated as `utf8mb4_bin`

- Fix the issue that the QPS panel for the TiFlash coprocessor executor always displays 0 in Grafana
 - Fix the wrong result of the `FROM_UNIXTIME` function when input is NULL
- Tools
 - TiCDC
 - * Fix the issue that TiCDC leaks memory in some cases [#942](#)
 - * Fix the issue that TiCDC might panic in Kafka sink [#912](#)
 - * Fix the issue that CommitTs or ResolvedTs (CRTs) might be less than `resolvedTs` in puller [#927](#)
 - * Fix the issue that `changefeed` might be blocked by MySQL driver [#936](#)
 - * Fix the incorrect Resolved Ts interval of TiCDC [#8573](#)
 - Backup & Restore (BR)
 - * Fix a panic that might occur during checksum [#479](#)
 - * Fix a panic that might occur after the change of PD Leader [#496](#)
 - Dumpling
 - * Fix the issue that the NULL value for the binary type is not handled properly [#137](#)
 - TiDB Lightning
 - * Fix the issue that all failed operations of writes and ingests are mistakenly displayed as successful [#381](#)
 - * Fix the issue that some checkpoint updates might not be written to the database before TiDB Lightning exits [#386](#)

14.7.12 TiDB 4.0.5 Release Notes

Release date: August 31, 2020

TiDB version: 4.0.5

14.7.12.1 Compatibility Changes

- TiDB
 - Change `drop partition` and `truncate partition`'s job arguments to support the ID array of multiple partitions [#18930](#)
 - Add the delete-only state for checking `add partition` replicas [#18865](#)

14.7.12.2 New Features

- TiKV
 - Define error code for errors [#8387](#)
- TiFlash
 - Support the unified log format with TiDB
- Tools
 - TiCDC
 - * Support Kafka SSL connection [#764](#)
 - * Support outputting the old value [#708](#)
 - * Add the column flags [#796](#)
 - * Support outputting the DDL statements and table schema of the previous version [#799](#)

14.7.12.3 Improvements

- TiDB
 - Optimize the performance of `DecodePlan` for big union queries [#18941](#)
 - Reduce the number of GC lock scans when the `Region cache miss` error occurs [#18876](#)
 - Ease the impact of statistical feedback on cluster performance [#18772](#)
 - Support canceling operations before the RPC response is returned [#18580](#)
 - Add the HTTP API to generate the TiDB metric profile [#18531](#)
 - Support scattering partitioned tables [#17863](#)
 - Add detailed memory usage of each instance in Grafana [#18679](#)
 - Show the detailed runtime information of the `BatchPointGet` operator in the result of `EXPLAIN` [#18892](#)
 - Show the detailed runtime information of the `PointGet` operator in the result of `EXPLAIN` [#18817](#)
 - Warn the potential deadlock for `Consume` in `remove()` [#18395](#)
 - Refine the behaviors of `StrToInt` and `StrToFloat` and support converting JSON to the `date`, `time`, and `timestamp` types [#18159](#)
 - Support limiting the memory usage of the `TableReader` operator [#18392](#)
 - Avoid too many times of backoff when retrying the `batch cop` request [#18999](#)
 - Improve compatibility for `ALTER TABLE` algorithms [#19270](#)
 - Make the single partitioned table support `IndexJoin` on the inner side [#19151](#)
 - Support searching the log file even when the log includes invalid lines [#18579](#)
- PD

- Support scattering Regions in stores with special engines (such as TiFlash) [#2706](#)
- Support the Region HTTP API to prioritize Region scheduling of a given key range [#2687](#)
- Improve the leader distribution after Region scattering [#2684](#)
- Add more tests and logs for the TSO request [#2678](#)
- Avoid invalid cache updates after the leader of a Region has changed [#2672](#)
- Add an option to allow `store.GetLimit` to return the tombstone stores [#2743](#)
- Support synchronizing the Region leader change between the PD leader and followers [#2795](#)
- Add commands for querying the GC safepoint service [#2797](#)
- Replace the `region.Clone` call in filters to improve performance [#2801](#)
- Add an option to disable updating Region flow cache to improve the performance of the large cluster [#2848](#)
- TiFlash
 - Add more Grafana panels to display metrics of CPU, I/O, RAM usages and metrics of the storage engine
 - Reduce I/O operations by optimizing the processing logic of Raft logs
 - Accelerate Region scheduling for the blocked `add partition` DDL statement
 - Optimize compactations of delta data in DeltaTree to reduce read and write amplification
 - Optimize the performance of applying Region snapshots by preprocessing the snapshots using multiple threads
 - Optimize the number of opening file descriptors when the read load of TiFlash is low to reduce system resource consumption
 - Optimize the number of unnecessary small files created when TiFlash restarts
 - Support encryption at rest for data storage
 - Support TLS for data transfer
- Tools
 - TiCDC
 - * Lower the frequency of getting TSO [#801](#)
 - Backup & Restore (BR)
 - * Optimize some logs [#428](#)
 - Dumpling
 - * Release FTWRL after connections are created to reduce the lock time for MySQL [#121](#)
 - TiDB Lightning
 - * Optimize some logs [#352](#)

14.7.12.4 Bug Fixes

- TiDB
 - Fix the `should ensure all columns have the same length` error that occurs because the `ErrTruncate/Overflow` error is incorrectly handled in the `builtinCastRealAsDecimalSig` function [#18967](#)
 - Fix the issue that the `pre_split_regions` table option does not work in the partitioned table [#18837](#)
 - Fix the issue that might cause a large transaction to be terminated prematurely [#18813](#)
 - Fix the issue that using the `collation` functions get wrong query results [#18735](#)
 - Fix the bug that the `getAutoIncrementID()` function does not consider the `tidb_snapshot` session variable, which might cause the dumper tool to fail with the `table not exist` error [#18692](#)
 - Fix the `unknown column` error for SQL statement like `select a from t ↩ having t.a` [#18434](#)
 - Fix the panic issue that writing the 64-bit unsigned type into the hash partitioned table causes overflow and gets an unexpected negative number when the partition key is the integer type [#18186](#)
 - Fix the wrong behavior of the `char` function [#18122](#)
 - Fix the issue that the `ADMIN REPAIR TABLE` statement cannot parse integer in the expressions on the range partition [#17988](#)
 - Fix the wrong behavior of the `SET CHARSET` statement [#17289](#)
 - Fix the bug caused by the wrong collation setting which leads to the wrong result of the `collation` function [#17231](#)
 - Fix the issue that `STR_TO_DATE`'s handling of the format tokens '`%r`', '`%h`' is inconsistent with that of MySQL [#18727](#)
 - Fix issues that the TiDB version information is inconsistent with that of PD/TiKV in the `cluster_info` table [#18413](#)
 - Fix the existent checks for pessimistic transactions [#19004](#)
 - Fix the issue that executing `union select for update` might cause concurrent race [#19006](#)
 - Fix the wrong query result when `apply` has a child of the `PointGet` operator [#19046](#)
 - Fix the incorrect result that occurs when `IndexLookUp` is in the inner side of the `Apply` operator [#19496](#)
 - Fix the incorrect result of `anti-semi-join` queries [#19472](#)
 - Fix the incorrect result caused by the mistaken usage of `BatchPointGet` [#19456](#)
 - Fix the incorrect result that occurs when `UnionScan` is in the inner side of the `Apply` operator [#19496](#)
 - Fix the panic caused by using the `EXECUTE` statement to print an expensive query log [#17419](#)
 - Fix the index join error when the join key is `ENUM` or `SET` [#19235](#)

- Fix the issue that the query range cannot be built when the `NULL` value exists on the index column [#19358](#)
 - Fix the data race issue caused by updating the global configuration [#17964](#)
 - Fix the panic issue occurs when modifying the character set in an uppercase schema [#19286](#)
 - Fix an unexpected error caused by changing the temporary directory during the disk spill action [#18970](#)
 - Fix the wrong hash key for the decimal type [#19131](#)
 - Fix the issue that the `PointGet` and `BatchPointGet` operators do not consider the partition selection syntax and get incorrect results [#19141](#)
 - Fix the incorrect results when using the `Apply` operator together with the `UnionScan` operator [#19104](#)
 - Fix the bug that causes the indexed virtual generated column to return wrong value [#17989](#)
 - Add the lock for runtime statistics to fix a panic caused by concurrent execution [#18983](#)
- TiKV
 - Speed up leader election when Hibernate Region is enabled [#8292](#)
 - Fix the memory leak issue during scheduling [#8357](#)
 - Add the `hibernate-timeout` configuration item to prevent the leader from becoming hibernate too fast [#8208](#)
 - PD
 - Fix the bug that the TSO request might fail at the time of leader change [#2666](#)
 - Fix the issue that sometimes Region replicas cannot be scheduled to the optimal state when placement rules are enabled [#2720](#)
 - Fix the issue that `Balance Leader` does not work when placement rules are enabled [#2726](#)
 - Fix the issue that unhealthy stores are not filtered from store load statistics [#2805](#)
 - TiFlash
 - Fix the issue that TiFlash cannot start normally after upgrading from an earlier version if the name of the database or table contains special characters
 - Fix the issue that the TiFlash process can not exit if any exceptions are thrown during initialization
 - Tools
 - Backup & Restore (BR)
 - * Fix the issue of duplicated calculation of total KV and total bytes in the backup summary log [#472](#)

- * Fix the issue that the import mode does not work in the first 5 minutes after switching to this mode [#473](#)
- Dumpling
 - * Fix the issue that FTWRL lock is not released in time [#128](#)
- TiCDC
 - * Fix the issue that the failed `changefeed` cannot be removed [#782](#)
 - * Fix invalid `delete` events by selecting one unique index as the handle index [#787](#)
 - * Fix the bug that GC safepoint is forwarded beyond the checkpoint of stopped `changefeed` [#797](#)
 - * Fix the bug that the network I/O waiting blocks tasks to exit [#825](#)
 - * Fix the bug that some unnecessary data might be mistakenly replicated to the downstream [#743](#)
- TiDB Lightning
 - * Fix the syntax error on empty binary/hex literals when using TiDB backend [#357](#)

14.7.13 TiDB 4.0.4 Release Notes

Release date: July 31, 2020

TiDB version: 4.0.4

14.7.13.1 Bug Fixes

- TiDB
 - Fix the issue of getting stuck when querying `information_schema.columns` [#18849](#)
 - Fix the errors that occur when the `PointGet` and `BatchPointGet` operators encounter `in null` [#18848](#)
 - Fix the wrong result of `BatchPointGet` [#18815](#)
 - Fix the issue of incorrect query result that occurs when the `HashJoin` operator encounters the `set` or `enum` type [#18859](#)

14.7.14 TiDB 4.0.3 Release Notes

Release date: July 24, 2020

TiDB version: 4.0.3

14.7.14.1 New Features

- TiDB Dashboard
 - Display detailed TiDB Dashboard version information [#679](#)
 - Show browser compatibility notice for unsupported browsers or outdated browsers [#654](#)
 - Support searching in the **SQL Statements** page [#658](#)
- TiFlash
 - Implement file encryption in TiFlash proxy
- Tools
 - Backup & Restore (BR)
 - * Support compressing backup files using zstd, lz4 or snappy [#404](#)
 - TiCDC
 - * Support configuring `kafka-client-id` in MQ sink-uri [#706](#)
 - * Support updating `changefeed` configuration offline [#699](#)
 - * Support setting customized `changefeed` name [#727](#)
 - * Support TLS and MySQL SSL connection [#347](#)
 - * Support outputting changes in the Avro format [#753](#)
 - * Support the Apache Pulsar sink [#751](#)
 - Dumpling
 - * Support the specialized CSV separator and delimiter [#116](#)
 - * Support specifying the format of the output file name [#122](#)

14.7.14.2 Improvements

- TiDB
 - Add the `tidb_log_desensitization` global variable to control whether to do desensitization when logging SQL queries [#18581](#)
 - Enable `tidb_allow_batch_cop` by default [#18552](#)
 - Speed up canceling a query [#18505](#)
 - Add a header for the `tidb_decode_plan` result [#18501](#)
 - Make the configuration checker compatible with earlier versions of the configuration file [#18046](#)
 - Enable collecting the execution information by default [#18518](#)
 - Add the `tiflash_tables` and `tiflash_segments` system tables [#18536](#)
 - Move `AUTO RANDOM` out of experimental features and announce its general availability. The improvements and compatibility changes are as follows:

- * Deprecate `experimental.allow-auto-random` in the configuration file. No matter how this item is configured, you can always define the `AUTO RANDOM` feature on columns. [#18613](#) [#18623](#)
- * Add the `tidb_allow_auto_random_explicit_insert` session variable to control the explicit writes on `AUTO RANDOM` columns. The default value is `false`. This is to avoid the unexpected `AUTO_RANDOM_BASE` update caused by explicit writes on columns. [#18508](#)
- * Allow defining `AUTO_RANDOM` only on `BIGINT` and `UNSIGNED BIGINT` columns and restrict the maximum number of shard bits to 15, which avoids the allocatable space being consumed too quickly [#18538](#)
- * Do not trigger the `AUTO_RANDOM_BASE` update when defining the `AUTO_RANDOM` ↗ attribute on the `BIGINT` column and inserting the negative value into the primary key [#17987](#)
- * Use the highest bit of an integer for ID allocation when defining the `AUTO_RANDOM` attribute on `UNSIGNED BIGINT` columns, which gets more allocable space [#18404](#)
- * Support updating the `AUTO_RANDOM` attribute in the result of `SHOW CREATE` ↗ `TABLE` [#18316](#)

- TiKV

- Introduce the new `backup.num-threads` configuration to control the size of the backup thread pool [#8199](#)
- Do not send store heartbeats when receiving snapshots [#8136](#)
- Support dynamically changing the shared block cache's capacity [#8232](#)

- PD

- Support the JSON formatted log [#2565](#)

- TiDB Dashboard

- Improve the Key Visualizer bucket merge for cold logical ranges [#674](#)
- Rename the configuration item of `disable-telemetry` to `enable-telemetry` for consistency [#684](#)
- Show the progress bar when switching pages [#661](#)
- Ensure that the slow log search now follows the same behavior as log search when there are space separators [#682](#)

- TiFlash

- Change the unit of the **DDL Jobs** panel in Grafana to `operations per minute`
- Add a new dashboard in Grafana to show more metrics about **TiFlash-Proxy**
- Reduce IOPS in TiFlash proxy

- Tools

- TiCDC
 - * Replace table ID with table name in metrics [#695](#)
- Backup & Restore (BR)
 - * Support outputting JSON logs [#336](#)
 - * Support enabling pprof during runtime [#372](#)
 - * Speed up DDL executions by sending DDL concurrently during restore [#377](#)
- TiDB Lightning
 - * Deprecate `black-white-list` with a newer and easier-to-understand filter format [#332](#)

14.7.14.3 Bug Fixes

- TiDB
 - Return an error instead of an empty set for `IndexHashJoin` when an error occurs during execution [#18586](#)
 - Fix the recurring panic when gRPC transportReader is broken [#18562](#)
 - Fix the issue that Green GC does not scan locks on offline stores which might cause data incompleteness [#18550](#)
 - Forbid processing a non-read-only statement using TiFlash engine [#18534](#)
 - Return the actual error message when a query connection panics [#18500](#)
 - Fix the issue that the `ADMIN REPAIR TABLE` execution fails to reload the table metadata on the TiDB node [#18323](#)
 - Fix the data inconsistency issue occurred because the lock of a written and deleted primary key in one transaction is resolved by another transaction [#18291](#)
 - Make spilling disk work well [#18288](#)
 - Fix the error reported when the `REPLACE INTO` statement works on the table that contains generated columns [#17907](#)
 - Return the OOM error when the `IndexHashJoin` and `IndexMergeJoin` workers panic [#18527](#)
 - Fix the bug that the execution of `Index Join` might return wrong results in special cases when the index used by `Index Join` contains the integer primary key [#18565](#)
 - Fix the issue that when the new collation is enabled on the cluster, the data updated on columns with the new collation in a transaction cannot be read through the unique index [#18703](#)
- TiKV
 - Fix the issue that reads might get stale data during merging [#8113](#)
 - Fix the issue that collation does not work on the `min/max` function when aggregation is pushed down to TiKV [#8108](#)

- PD
 - Fix the issue that creating TSO stream might be blocked for a while if the server crashes [#2648](#)
 - Fix the issue that `getSchedulers` might cause a data race [#2638](#)
 - Fix the issue that deleting the scheduler might cause deadlocks [#2637](#)
 - Fix the bug that placement rules are not considered when `balance-leader-→ scheduler` is enabled [#2636](#)
 - Fix the issue that sometimes service `safepoint` cannot be set properly, which might make BR and dumpling fail [#2635](#)
 - Fix the issue that the target store in `hot region scheduler` is incorrectly selected [#2627](#)
 - Fix the issue that the TSO request might take too long when PD leader is switched [#2622](#)
 - Fix the issue of stale scheduler after leader change [#2608](#)
 - Fix the issue that sometimes replicas of a Region cannot be adjusted to the best location when placement rules are enabled [#2605](#)
 - Fix the issue that the deployment path of the store is not updated according to the change of deployment directory [#2600](#)
 - Prevent `store limit` from changing to zero [#2588](#)
- TiDB Dashboard
 - Fix the TiDB connection error when TiDB is scaled out [#689](#)
 - Fix the issue that TiFlash instances are not displayed in the log searching page [#680](#)
 - Fix the issue of metric selection reset after refreshing the overview page [#663](#)
 - Fix a connection issue in some TLS scenarios [#660](#)
 - Fix the issue that the language dropdown box is not displayed correctly in some cases [#677](#)
- TiFlash
 - Fix the issue that TiFlash crashes after renaming the primary key column
 - Fix the issue that concurrent `Learner Read` and `Remove Region` might cause deadlocks
- Tools
 - TiCDC
 - * Fix the issue that TiCDC leaks memory in some cases [#704](#)
 - * Fix the issue that unquoted table name causes the SQL syntax error [#676](#)
 - * Fix the issue that the processor does not fully exit after `p.stop` is called [#693](#)
 - Backup & Restore (BR)
 - * Fix the issue that the backup time might be negative [#405](#)

- Dumpling
 - * Fix the issue that Dumpling omits the NULL value when `--r` is specified [#119](#)
 - * Fix the bug that flushing tables might not work for tables to dump [#117](#)
- TiDB Lightning
 - * Fix the issue that `--log-file` does not take effect [#345](#)
- TiDB Binlog
 - * Fix the issue that when TiDB Binlog replicates data to the downstream with TLS enabled, Drainer cannot be started which occurs because TLS is not enabled on the database driver used to update the checkpoint [#988](#)

14.7.15 TiDB 4.0.2 Release Notes

Release date: July 1, 2020

TiDB version: 4.0.2

14.7.15.1 Compatibility Changes

- TiDB
 - Remove sensitive information in the slow query log and the statement summary table [#18130](#)
 - Forbid negative value in the sequence cache [#18103](#)
 - Remove tombstone TiKV and TiFlash stores from the `CLUSTER_INFO` table [#17953](#)
 - Change the diagnostic rule from `current-load` to `node-check` [#17660](#)
- PD
 - Persist `store-limit` and remove `store-balance-rate` [#2557](#)

14.7.15.2 New Change

- By default, TiDB and TiDB Dashboard share usage details with PingCAP to help understand how to improve the product [#18180](#). For details about what is shared and how to disable the sharing, see [Telemetry](#).

14.7.15.3 New Features

- TiDB
 - Support the `MEMORY_QUOTA()` hint in `INSERT` statements [#18101](#)
 - Support authentication based on the `SAN` field of TLS certificate [#17698](#)
 - Support collation for the `REGEXP()` function [#17581](#)
 - Support the `sql_select_limit` session and global variable [#17604](#)
 - Support splitting the Region for the newly added partition by default [#17665](#)
 - Support pushing the `IF()`/`BITXOR()`/`BITNEG()`/`JSON_LENGTH()` functions to the TiFlash Coprocessor [#17651](#) [#17592](#)
 - Support a new aggregate function `APPROX_COUNT_DISTINCT()` to calculate the approximate result of `COUNT(DISTINCT)` [#18120](#)
 - Support collation in TiFlash and pushing collation-related functions to TiFlash [#17705](#)
 - Add the `STATUS_ADDRESS` column in the `INFORMATION_SCHEMA.INSPECTION_RESULT` table to indicate the status address of servers [#17695](#)
 - Add the `SOURCE` column in the `MYSQL.BIND_INFO` table to indicate the how the bindings are created [#17587](#)
 - Add the `PLAN_IN_CACHE` and `PLAN_CACHE_HITS` columns in the `PERFORMANCE_SCHEMA.EVENTS_STATEMENTS_SUMMARY_BY_DIGEST` table to indicate the plan cache usage of SQL statements [#17493](#)
 - Add the `enable-collect-execution-info` configuration item and the `tidb_enable_collect_execution_info` session variable to control whether to collect execution information of each operator and record the information in the slow query log [#18073](#) [#18072](#)
 - Add the `tidb_slow_log_masking` global variable to control whether to desensitize the queries in slow query log [#17694](#)
 - Add a diagnostic rule in the `INFORMATION_SCHEMA.INSPECTION_RESULT` table for the `storage.block-cache.capacity` TiKV configuration item [#17671](#)
 - Add the `BACKUP` and `RESTORE` SQL statements to back up and restore data [#15274](#)
- TiKV
 - Support the `encryption-meta` command in TiKV Control [#8103](#)
 - Add a perf context metric for `RocksDB::WriteImpl` [#7991](#)
- PD
 - Support the operator to fail immediately when trying to remove a leader peer [#2551](#)
 - Set a suitable default store limit for TiFlash stores [#2559](#)
- TiFlash
 - Support new aggregation function `APPROX_COUNT_DISTINCT` in Coprocessor

- Enable the `rough set filter` feature by default
 - Enable TiFlash to run on the ARM architecture
 - Support pushing down the `JSON_LENGTH` function in Coprocessor
- Tools
 - TiCDC
 - * Support migrating sub-tasks to new captures [#665](#)
 - * Add a `cli` command to delete the TiCDC GC TTL [#652](#)
 - * Support canal protocol in MQ sink [#649](#)

14.7.15.4 Improvements

- TiDB
 - Reduce the query latency caused by the Golang memory allocation when CM-Sketch consumes too much memory [#17545](#)
 - Reduce the QPS recovery duration of a cluster when a TiKV server is in the failure recovery process [#17681](#)
 - Support pushing aggregate functions to TiKV/TiFlash Coprocessor on partition tables [#17655](#)
 - Improve the accuracy of row count estimation for index equal conditions [#17611](#)
- TiKV
 - Improve the PD client panic log [#8093](#)
 - Add back the `process_cpu_seconds_total` and `process_start_time_seconds` monitoring metrics [#8029](#)
- TiFlash
 - Improve backward compatibility when upgrading from an older version [#786](#)
 - Reduce memory consumption of delta index [#787](#)
 - Use the more efficient update algorithm for delta index [#794](#)
- Tools
 - Backup & Restore (BR)
 - * Improve the performance by pipelining the restore process [#266](#)

14.7.15.5 Bug Fixes

- TiDB
 - Fix the issue of incorrect execution plan obtained from the plan cache after `tidb_isolation_read_engines` is changed [#17570](#)
 - Fix the occasional runtime error that occurs when executing the EXPLAIN FOR → CONNECTION statement [#18124](#)
 - Fix the incorrect result of the `last_plan_from_cache` session variable in some cases [#18111](#)
 - Fix the runtime error that occurs when executing the `UNIX_TIMESTAMP()` function from the plan cache [#18002](#) [#17673](#)
 - Fix the runtime error when the child of `HashJoin` executor returns the NULL column [#17937](#)
 - Fix the runtime error caused by concurrently executing the `DROP DATABASE` statement and other DDL statements in the same database [#17659](#)
 - Fix the incorrect result of the `COERCIBILITY()` function on user variables [#17890](#)
 - Fix the issue that the `IndexMergeJoin` executor occasionally gets stuck [#18091](#)
 - Fix the hang issue of the `IndexMergeJoin` executor when out of memory quota and query cancelling is triggered [#17654](#)
 - Fix the excessive counting memory usage of the `Insert` and `Replace` executors [#18062](#)
 - Fix the issue that the data replication to TiFlash storage is stopped when `DROP` → `DATABASE` and `DROP TABLE` are executed concurrently in the same database [#17901](#)
 - Fix the BACKUP/RESTORE failure between TiDB and the object storage service [#17844](#)
 - Fix the incorrect error message of privilege check failure when access is denied [#17724](#)
 - Discard the query feedbacks generated from the `DELETE/UPDATE` statement [#17843](#)
 - Forbid altering `AUTO_RANDOM_BASE` for a table without `AUTO_RANDOM` property [#17828](#)
 - Fix the issue that the `AUTO_RANDOM` column is allocated wrong results when the table is moved between databases by `ALTER TABLE ... RENAME` [#18243](#)
 - Fix the issue that some system tables cannot be accessed when setting the value of `tidb_isolation_read_engines` without `tidb` [#17719](#)
 - Fix the inaccurate result of JSON comparison on large integers and float values [#17717](#)
 - Fix the incorrect decimal property for the result of the `COUNT()` function [#17704](#)
 - Fix the incorrect result of the `HEX()` function when the type of input is the binary string [#17620](#)
 - Fix the issue that an empty result is returned when querying the `INFORMATION_SCHEMA` → `.INSPECTION_SUMMARY` table without filter condition [#17697](#)

- Fix the issue that the hashed password used by the `ALTER USER` statement to update user information is unexpected [#17646](#)
- Support collation for `ENUM` and `SET` values [#17701](#)
- Fix the issue that the timeout mechanism for pre-splitting Regions does not work when creating a table [#17619](#)
- Fix the issue that the schema is unexpectedly updated when a DDL job is retried, which might break the atomicity of DDL jobs [#17608](#)
- Fix the incorrect result of the `FIELD()` function when the argument contains the column [#17562](#)
- Fix the issue that the `max_execution_time` hint does not work occasionally [#17536](#)
- Fix the issue that the concurrency information is redundantly printed in the result of `EXPLAIN ANALYZE` [#17350](#)
- Fix the incompatible behavior of `%h` on the `STR_TO_DATE` function [#17498](#)
- Fix the issue that the follower/learner keeps retrying when `tidb_replica_read` is set to `follower` and there is a network partition between the leader and the follower/learner [#17443](#)
- Fix the issue that TiDB sends too many pings to PD follower in some cases [#17947](#)
- Fix the issue that the range partition table of older versions cannot be loaded in TiDB v4.0 [#17983](#)
- Fix the SQL statement timeout issue when multiple Region requests fail at the same time by assigning different `Backoffer` for each Region [#17585](#)
- Fix the MySQL incompatible behavior when parsing `DateTime` delimiters [#17501](#)
- Fix the issue that TiKV requests are occasionally sent to the TiFlash server [#18105](#)
- Fix the data inconsistency issue occurred because the lock of a written and deleted primary key in one transaction is resolved by another transaction [#18250](#)

- TiKV

- Fix a memory safety issue for the status server [#8101](#)
- Fix the issue of lost precision in JSON numeric comparison [#8087](#)
- Fix the wrong query slow log [#8050](#)
- Fix the issue that a peer cannot be removed when its store is isolated during multiple merge processes [#8048](#)
- Fix the issue that `tikv-ctl recover-mvcc` does not remove invalid pessimistic locks [#8047](#)
- Fix the issue that some Titan histogram metrics are missing [#7997](#)
- Fix the issue that TiKV returns duplicated error to TiCDC [#7887](#)

- PD

- Check the correctness of the `pd-server.dashboard-address` configuration item [#2517](#)

- Fix the panic issue of PD when setting `store-limit-mode` to `auto` [#2544](#)
 - Fix the issue that hotspots cannot be identified in some cases [#2463](#)
 - Fix the issue that placement rules prevent the store from changing to `tombstone` in some cases [#2546](#)
 - Fix the panic issue of PD when upgrading from earlier versions in some cases [#2564](#)
- TiFlash
 - Fix the issue that the proxy might panic when the `region not found` error occurs
 - Fix the issue that the I/O exception thrown in `drop table` might lead to synchronization failure of TiFlash schema

14.7.16 TiDB 4.0.1 Release Notes

Release date: June 12, 2020

TiDB version: 4.0.1

14.7.16.1 New Features

- TiKV
 - Add the `--advertise-status-addr` start flag to specify the status address to advertise [#8046](#)
- PD
 - Support the internal proxy for the built-in TiDB Dashboard [#2511](#)
 - Support setting a custom timeout for PD client [#2509](#)
- TiFlash
 - Support the TiDB new collation framework
 - Support pushing down the `If/BitAnd/BitOr/BitXor/BitNot/Json_length` functions to TiFlash
 - Support the Resolve Lock logic for large transactions in TiFlash
- Tools
 - Backup & Restore (BR)
 - * Add a version check when starting BR to avoid the issue that BR and the TiDB cluster are incompatible [#311](#)

14.7.16.2 Bug Fixes

- TiKV
 - Fix the issue that the `use-unified-pool` configuration in the startup log is incorrectly printed [#7946](#)
 - Fix the issue that the tikv-ctl does not support relative path [#7963](#)
 - Fix the bug that the monitoring metric of Point Selects is inaccurate [#8033](#)
 - Fix the issue that a peer might not be destroyed after the network isolation disappears [#8006](#)
 - Fix the issue that a request for read index might get outdated commit index [#8043](#)
 - Improve the reliability of backup and restore with S3 and GCS storages [#7917](#)
- PD
 - Prevent misconfiguration of Placement Rules in some situations [#2516](#)
 - Fix the issue that deleting the Placement Rule might cause panic [#2515](#)
 - Fix a bug that the store information cannot be obtained when the store's used size is zero [#2474](#)
- TiFlash
 - Fix the issue that default value of the `bit` type column in TiFlash is incorrectly parsed
 - Fix the miscalculation of 1970-01-01 00:00:00 UTC in some timezones in TiFlash

14.7.17 TiDB 4.0 GA Release Notes

Release date: May 28, 2020

TiDB version: 4.0.0

14.7.17.1 Compatibility Changes

- TiDB
 - Optimize the error message of the large-sized transactions for easier troubleshooting [#17219](#)
- TiCDC
 - Optimize the structure of the `Changefeed` configuration file to improve usability [#588](#)
 - Add the `ignore-txn-start-ts` configuration item, and change the condition from `commit_ts` to `start_ts` during transactions filtering [#589](#)

14.7.17.2 Important Bug Fixes

- TiKV
 - Fix the `DefaultNotFound` error that occurs when backing up using `Backup & Restore` (BR) [#7937](#)
 - Fix system panics caused by the out-of-order `ReadIndex` packages [#7930](#)
 - Fix system panics caused by incorrectly removing snapshot files after TiKV is restarted [#7927](#)
- TiFlash
 - Fix the possible data loss issue that occurs when the system panics because of incorrect processing logic of `Raft Admin Command`

14.7.17.3 New Features

- TiDB
 - Add the `committer-concurrency` configuration item to control the number of `goroutines` in the retry-commit phase [#16849](#)
 - Support the `show table partition regions` syntax [#17294](#)
 - Add the `tmp-storage-quota` configuration item to limit the temporary disk space used by the TiDB server [#15700](#)
 - Support checking whether the partitioned table uses a unique prefix index when creating and changing tables [#17213](#)
 - Support the `insert/replace into tbl_name partition(partition_name_list ↳)` statement [#17313](#)
 - Support checking the value of `collations` when using the `Distinct` function [#17240](#)
 - Support the `is null` filter condition during the Hash partition pruning [#17310](#)
 - Support `admin check index`, `admin cleanup index`, and `admin recover ↳ index` in partitioned tables [#17392](#) [#17405](#) [#17317](#)
 - Support range partition pruning for the `in` expressions [#17320](#)
- TiFlash
 - Support filtering out the data corresponding to the qualified TSO through the `min commit ts` value of the `Lock CF` when the Learner reads the data
 - Add the feature that the system explicitly reports an error to avoid incorrect calculation results when the value of `TIMESTAMP` type is less than `1970-01-01 ↳ 00:00:00`
 - Support using flags in regular expressions when searching logs
- TiKV
 - Support collation rules of `ascii_bin` and `latin1_bin` encoding [#7919](#)

- PD
 - Support specifying the reverse proxy resource prefix for built-in TiDB Dashboard [#2457](#)
 - Support returning the `pending peer` and `down peer` information in interfaces of the PD client Region [#2443](#)
 - Add monitoring items such as `Direction of hotspot move leader`, `Direction ↗ of hotspot move peer`, and `Hot cache read entry number` [#2448](#)
- Tools
 - Backup & Restore (BR)
 - * Support the backup and restore of `Sequence` and `View` [#242](#)
 - TiCDC
 - * Support checking the validity of `Sink URI` when creating `Changefeed` [#561](#)
 - * Support checking whether the PD and TiKV versions meet the system requirements during system startup [#570](#)
 - * Support scheduling multiple tables in the same scheduling task generation cycle [#572](#)
 - * Add information about node roles in HTTP API [#591](#)

14.7.17.4 Bug Fixes

- TiDB
 - Fix the issue of unexpected timeouts when sending and receiving messages by disabling TiDB to send batch commands to TiFlash [#17307](#)
 - Fix the issue of incorrectly distinguishing signed and unsigned integers during partition pruning, which improves performance [#17230](#)
 - Fix the issue of upgrade failure from v3.1.1 to v4.0 because of the incompatible `mysql.user` table [#17300](#)
 - Fix the issue of incorrect selection of the partition in the `update` statement [#17305](#)
 - Fix system panics when receiving an unknown error message from TiKV [#17380](#)
 - Fix system panics caused by incorrect processing logic when creating the table that is `key` partitioned [#17242](#)
 - Fix the issue that the wrong `Index Merge Join` plan is selected because of incorrect optimizer processing logic [#17365](#)
 - Fix the issue of inaccurate `duration` monitoring metric of the `SELECT` statement in Grafana [#16561](#)
 - Fix the issue that the GC worker is blocked when the system error occurs [#16915](#)
 - Fix the issue that the `UNIQUE` constraint on a boolean column results in an incorrect result in a comparison [#17306](#)
 - Fix system panics caused by incorrect processing logic when `tidb_opt_agg_push_down ↗` is enabled and the aggregation function pushes down the partitioned table [#17328](#)

- Fix the issue of accessing failed TiKV nodes in some cases [#17342](#)
- Fix the issue that the `isolation-read` configuration item in `tidb.toml` does not take effect [#17322](#)
- Fix the issue of incorrect order of output results due to incorrect processing logic when `hint` is used to enforce the stream aggregation [#17347](#)
- Fix the behavior that `insert` processes DIV under different `SQL_MODE` [#17314](#)
- TiFlash
 - Fix the issue that the matching behavior of regular expressions in the search log feature is inconsistent with other components
 - Fix the issue of excessive restart time when nodes write large amounts of data by disabling the delay processing optimization of Raft Compact Log Command by default
 - Fix the issue that the system fails to start because TiDB incorrectly processes the `DROP DATABASE` statement in some scenarios
 - Fix the issue that the method of collecting CPU information in `Server_info` is different from that in other components
 - Fix the issue that the error `Too Many Pings` is reported when the `Query` statement is executed if `batch coprocessor` is enabled
 - Fix the issue that Dashboard fails to display the correct `deploy path` information because TiFlash does not report the related information
- TiKV
 - Fix the `DefaultNotFound` error that occurs when backing up using BR [#7937](#)
 - Fix system panics caused by out-of-order `ReadIndex` packets [#7930](#)
 - Fix the issue that an unexpected error is returned because the read request callback function is not called [#7921](#)
 - Fix system panics caused by incorrectly removing snapshot files when TiKV is restarted [#7927](#)
 - Fix the issue that the `master` key cannot be rotated due to incorrect processing logic in storage encryption [#7898](#)
 - Fix the issue that the received `lock cf` file of the snapshot is not encrypted when the storage encryption is enabled [#7922](#)
- PD
 - Fix the 404 error when deleting `evict-leader-scheduler` or `grant-leader-→ scheduler` using pd-ctl [#2446](#)
 - Fix the issue that the `presplit` feature might not work properly when the TiFlash replica exists [#2447](#)
- Tools
 - Backup & Restore (BR)

- * Fix the issue that the data restoration fails due to network issues when BR restores data from cloud storage [#298](#)
- TiCDC
 - * Fix system panics caused by data race [#565](#) [#566](#)
 - * Fix resource leaks or system blockages caused by incorrect processing logic [#574](#) [#586](#)
 - * Fix the issue that the command line gets stuck because CLI cannot connect to PD [#579](#)

14.7.18 TiDB 4.0 RC.2 Release Notes

Release date: May 15, 2020

TiDB version: 4.0.0-rc.2

14.7.18.1 Compatibility Changes

- TiDB
 - Remove the size limit for a single transaction (100 MB) when TiDB Binlog is enabled. Now the size limit for a transaction is 10 GB. However, if TiDB Binlog is enabled and the downstream is Kafka, configure the `txn-total-size-limit` parameter according to the message size limit of 1 GB in Kafka [#16941](#)
 - Change the behavior from querying the default time range to returning an error and requesting a specified time range if the time range is not specified when querying the `CLUSTER_LOG` table [#17003](#)
 - If the unsupported `sub-partition` or `linear hash` option is specified when creating the partitioned table using the `CREATE TABLE` statement, the normal table is created rather than the partitioned table with the options ignored [#17197](#)
- TiKV
 - Move the encryption-related configuration to the security-related configuration, which means changing `[encryption]` in the TiKV configuration file to `[security → .encryption]` [#7810](#)
- Tools
 - TiDB Lightning
 - * Change the default SQL mode to `ONLY_FULL_GROUP_BY,NO_AUTO_CREATE_USER →` when importing data to improve compatibility [#316](#)
 - * Disallow accessing the PD or TiKV port in the tidb-backend mode [#312](#)
 - * Print the log information to the tmp file by default, and print the path of the tmp file when TiDB Lightning is started [#313](#)

14.7.18.2 Important Bug Fixes

- TiDB
 - Fix the issue that the wrong partition is chosen when the WHERE clause has only one equivalent condition [#17054](#)
 - Fix the issue of wrong results caused by building the incorrect Index range when the WHERE clause only contains the string column [#16660](#)
 - Fix the panic issue that occurs when executing the PointGet query in the transaction after the DELETE operation [#16991](#)
 - Fix the issue that the GC worker might encounter the deadlock when an error occurs [#16915](#)
 - Avoid the unnecessary RegionMiss retry when the TiKV response is slow but not down [#16956](#)
 - Change the log level in the client in the handshake phase of the MySQL protocol to DEBUG to solve the problem that interferes with log output [#16881](#)
 - Fix the issue that the Region is not pre-split according to the PRE_SPLIT_REGIONS information defined by the table after the TRUNCATE operation [#16776](#)
 - Fix the issue of soaring goroutine caused by retry when TiKV is unavailable during the second phase of the two-phase commit [#16876](#)
 - Fix the panic issue of statement execution when some expressions cannot be pushed down [#16869](#)
 - Fix the wrong execution result of the IndexMerge operation on the partitioned table [#17124](#)
 - Fix the performance reduction of wide_table caused by the mutex contention of Memory Trackers [#17234](#)
- TiFlash
 - Fix the issue that the system cannot start normally after the upgrade if the name of the database or table contains special characters

14.7.18.3 New Features

- TiDB
 - Add support for the BACKUP and RESTORE commands to back up and restore data [#16960](#)
 - Support pre-checking the data volume in a single Region before commit and pre-splitting the Region when the data volume exceeds the threshold [#16959](#)
 - Add the new LAST_PLAN_FROM_CACHE variable with a Session scope to indicate whether the last executed statement hits the plan cache [#16830](#)
 - Support recording the Cop_time information in slow log and the SLOW_LOG table [#16904](#)

- Add in Grafana more metrics that monitor the memory status of Go Runtime [#16928](#)
- Support outputting the `forUpdateTS` and Read Consistency isolation level information in General Log [#16946](#)
- Support collapsing duplicate requests of resolving locks in TiKV Region [#16925](#)
- Support using the `SET CONFIG` statement to modify the configuration of PD/TiKV nodes [#16853](#)
- Support the `auto_random` option in the `CREATE TABLE` statement [#16813](#)
- Allocate TaskID for the DistSQL request to help TiKV better schedule and process requests [#17155](#)
- Support displaying the version information of the TiDB server after logging into the MySQL client [#17187](#)
- Support the `ORDER BY` clause in the `GROUP_CONCAT` function [#16990](#)
- Support displaying the `Plan_from_cache` information in slow log to indicate whether the statement hits plan cache [#17121](#)
- Add the feature that TiDB Dashboard can display the capacity information of TiFlash multi-disk deployment
- Add the feature of querying the TiFlash log using SQL statements in Dashboard
- TiKV
 - Support encryption debugging for tikv-ctl, so that tikv-ctl can be used to operate and manage the cluster when the encryption storage is enabled [#7698](#)
 - Support encrypting the lock column family in snapshots [#7712](#)
 - Use the heatmap in the Grafana dashboard for Raftstore latency summary to better diagnose the jitter issue [#7717](#)
 - Support setting the upper limit for the size of the gRPC message [#7824](#)
 - Add in Grafana dashboard the encryption-related monitoring metrics [#7827](#)
 - Support Application-Layer Protocol Negotiation (ALPN) [#7825](#)
 - Add more statistics about Titan [#7818](#)
 - Support using the task ID provided by the client as the identifier in the unified read pool to avoid that the priority of a task is lowered by another task in the same transaction [#7814](#)
 - Improve the performance of the `batch insert` request [#7718](#)
- PD
 - Eliminate the speed limit of removing peers when making a node offline [#2372](#)
- TiFlash
 - Change the name of the Count graph of **Read Index** in Grafana to **Ops**
 - Optimize the data for opening file descriptors when the system load is low to reduce system resource consumption
 - Add the capacity-related configuration parameter to limit the the data storage capacity

- Tools
 - TiDB Lightning
 - * Add the `fetch-mode` sub-command in `tidb-lightning-ctl` to print the TiKV cluster mode [#287](#)
 - TiCDC
 - * Support managing the replication task by using `cdc cli` (changefeed) [#546](#)
 - Backup & Restore (BR)
 - * Support automatically adjusting GC time during backup [#257](#)
 - * Adjust PD parameters when restoring data to speed up the restoration [#198](#)

14.7.18.4 Bug Fixes

- TiDB
 - Improve the logic that determines whether to use vectorization for expression execution in multiple operators [#16383](#)
 - Fix the issue that the `IndexMerge` hint fails to check the database name correctly [#16932](#)
 - Forbid truncating the sequence object [#17037](#)
 - Fix the issue that the `INSERT/UPDATE/ANALYZE/DELETE` statements can be performed on a sequence object [#16957](#)
 - Fix the issue that the internal SQL statements in the bootstrap phase are not correctly marked as internal queries in the Statement Summary table [#17062](#)
 - Fix the error that occurs when a filter condition supported by TiFlash but not by TiKV is pushed down to the `IndexLookupJoin` operator [#17036](#)
 - Fix the concurrency issue of the `LIKE` expression that might occur after the collation is enabled [#16997](#)
 - Fix the issue that the `LIKE` function cannot correctly build the `Range` query index after the collation is enabled [#16783](#)
 - Fix the issue that a wrong value is returned when executing `@@LAST_PLAN_FROM_CACHE` → after the `Plan Cache` statement is triggered [#16831](#)
 - Fix the issue that `TableFilter` on the index is missed when calculating candidate paths for `IndexMerge` [#16947](#)
 - Fix the issue that a physical query plan cannot be generated when using the `MergeJoin` hint and the `TableDual` operator exists [#17016](#)
 - Fix the wrong capitalization of the values in the `Stmt_Type` column of the Statement Summary table [#17018](#)
 - Fix the issue that the `Permission Denied` error is reported because the service cannot be started when different users use the same `tmp-storage-path` [#16996](#)
 - Fix the issue that the `NotNullFlag` result type is incorrectly set for an expression whose result type is determined by multiple input columns, such as `CASE WHEN #16995`

- Fix the issue that the green GC might leave unresolved locks when dirty stores exist [#16949](#)
 - Fix the issue that the green GC might leave unresolved locks when encountering a single key with multiple different locks [#16948](#)
 - Fix the issue of inserting a wrong value in the `INSERT VALUE` statement because a sub-query refers to a parent query column [#16952](#)
 - Fix the issue of incorrect results when using the `AND` operator on the `Float` value [#16666](#)
 - Fix the wrong information of the `WAIT_TIME` field in the expensive log [#16907](#)
 - Fix the issue that the `SELECT FOR UPDATE` statement cannot be recorded in the slow log in the pessimistic transaction mode [#16897](#)
 - Fix the wrong result that occurs when executing `SELECT DISTINCT` on a column of the `Enum` or `Set` type [#16892](#)
 - Fix the display error of `auto_random_base` in the `SHOW CREATE TABLE` statement [#16864](#)
 - Fix the incorrect value of `string_value` in the `WHERE` clause [#16559](#)
 - Fix the issue that the error message of the `GROUP BY` window function is inconsistent with that of MySQL [#16165](#)
 - Fix the issue that the `FLASH TABLE` statement fails to execute when the database name contains the uppercase letter [#17167](#)
 - Fix the inaccurate memory tracing of the Projection executor [#17118](#)
 - Fix the issue of incorrect time filtering of the `SLOW_QUERY` table in different time zones [#17164](#)
 - Fix the panic issue that occurs when `IndexMerge` is used with the virtual generated column [#17126](#)
 - Fix the capitalization issue of the `INSTR` and `LOCATE` function [#17068](#)
 - Fix the issue that the `tikv server timeout` error is reported frequently after the `tidb_allow_batch_cop` configuration is enabled [#17161](#)
 - Fix the issue that the result of performing `XOR` operation on the `Float` type is inconsistent with that of MySQL 8.0 [#16978](#)
 - Fix the issue that no error is reported when the unsupported `ALTER TABLE ↪ REORGANIZE PARTITION` statement is executed [#17178](#)
 - Fix the issue that an error is reported when `EXPLAIN FORMAT="dot" FOR ↪ CONNECTION ID` encounters an unsupported plan [#17160](#)
 - Fix the record issue of the prepared statement in the `EXEC_COUNT` column of the Statement Summary table [#17086](#)
 - Fix the issue that the value is not validated when setting the Statement Summary system variable [#17129](#)
 - Fix the issue that an error is reported if an overflow value is used to query the `UNSIGNED BIGINT` primary key when the plan cache is enabled [#17120](#)
 - Fix the incorrect QPS display by the machine instance and request type on the Grafana **TiDB Summary** dashboard [#17105](#)
- TiKV

- Fix the issue that many empty Regions are generated after restoration [#7632](#)
- Fix the panic issue of Raftstore when receiving out-of-order read index responses [#7370](#)
- Fix the issue that an invalid storage or coprocessor read pool configuration might not be rejected when the unified thread pool is enabled [#7513](#)
- Fix the panic issue of the `join` operation when the TiKV server is shut down [#7713](#)
- Fix the issue that no result is returned when searching TiKV slow logs via diagnostics API [#7776](#)
- Fix the issue that notable memory fragmentation is generated when the TiKV node is running for a long time [#7556](#)
- Fix the issue that the SQL statement fails to execute when an invalid date is stored [#7268](#)
- Fix the issue that the backup data cannot be restored from GCS [#7739](#)
- Fix the issue that KMS key ID is not validated during encryption at rest [#7719](#)
- Fix the underlying correctness issue of the Coprocessor in compilers of different architecture [#7714](#) [#7730](#)
- Fix the `snapshot ingestion` error when encryption is enabled [#7815](#)
- Fix the `Invalid cross-device link` error when rewriting the configuration file [#7817](#)
- Fix the issue of wrong toml format when writing the configuration file to an empty file [#7817](#)
- Fix the issue that a destroyed peer in Raftstore can still process requests [#7836](#)

- PD

- Fix the 404 issue that occurs when using the `region key` command in pd-ctl [#2399](#)
- Fix the issue that the monitor metrics of TSO and ID allocation are missing from the Grafana dashboard [#2405](#)
- Fix the issue that pd-recover is not included in the Docker image [#2406](#)
- Parse the path of data directory to an absolute path to fix the issue that TiDB Dashboard might not correctly display PD information [#2420](#)
- Fix the issue that there is no default output when using the `scheduler config`
→ `shuffle-region-scheduler` command in pd-ctl [#2416](#)

- TiFlash

- Fix the issue that the wrong information of used capacity is reported in some scenarios

- Tools

- TiDB Binlog
 - * Fix the issue that data of the `mediumint` type is not processed when the downstream is Kafka [#962](#)

- * Fix the issue that the reproto fails to parse the DDL statement when the database name in DDL is a keyword [#961](#)
- TiCDC
 - * Fix the issue of using the wrong time zone when the TZ environment variable is not set [#512](#)
 - * Fix the issue that the owner does not clean up the resources when the server exits because some errors are not handled correctly [#528](#)
 - * Fix the issue that TiCDC might be stuck when reconnecting to TiKV [#531](#)
 - * Optimize the memory usage when initializing the table schema [#534](#)
 - * Use the `watch` mode to monitor the replication status changes and perform quasi-real-time updates to reduce replication delay [#481](#)
- Backup & Restore (BR)
 - * Fix the issue that inserting data might trigger the `duplicate entry` error after BR restores a table with the `auto_random` attribute [#241](#)

14.7.19 TiDB 4.0 RC.1 Release Notes

Release date: April 28, 2020

TiDB version: 4.0.0-rc.1

14.7.19.1 Compatibility Changes

- TiKV
 - Disable the Hibernate Region feature by default [#7618](#)
- TiDB Binlog
 - Support the sequence DDL operation in Drainer [#950](#)

14.7.19.2 Important Bug Fixes

- TiDB
 - Fix the issue that the `INSERT ... ON DUPLICATE UPDATE` statement might be incorrectly executed on multiple rows in an explicit transaction because `MemBuffer` is not checked [#16689](#)
 - Fix the data inconsistency when locking duplicated keys on multiple rows [#16769](#)
 - Fix the panic that occurs when recycling the non-superbatch idle connection between TiDB instances [#16303](#)
- TiKV

- Fix the deadlock issue caused by the probe request from TiDB [#7540](#)
- Fix the issue that the minimum commit timestamp of a transaction might overflow which affects data correctness [#7638](#)
- TiFlash
 - Fix the data loss issue caused by the `rename table` operation when multiple data paths are configured
 - Fix the issue that an error occurs when reading data from a merged Region
 - Fix the issue that an error occurs when reading data from a Region that is in the abnormal state
 - Modify the mapping of table names in TiFlash to correctly support `recover ↪ table/flashback table`
 - Modify the storage path to fix the potential data loss issue that occurs when renaming the table
 - Fix the potential panic of TiDB when Super Batch is enabled
 - Modify the read mode in the online update scenario to improve the read performance
- TiCDC
 - Fix the replication failure that occurs because the schema internally maintained in TiCDC fails to correctly handle the timing issue of read and write operations [#438](#) [#450](#) [#478](#) [#496](#)
 - Fix the bug that the TiKV client fails to correctly maintain the internal resources when encountering some TiKV anomalies [#499](#) [#492](#)
 - Fix the bug that meta data is not correctly cleaned up and abnormally remains in the TiCDC nodes [#488](#) [#504](#)
 - Fix the issue that the TiKV client fails to correctly handle the repeated sending of the prewrite event [#446](#)
 - Fix the issue that the TiKV client fails to correctly handle the redundant prewrite events received before the initialization [#448](#)
- Backup & Restore (BR)
 - Fix the issue that checksum is still executed when checksum is disabled [#223](#)
 - Fix the incremental replication failure when `auto-random` or `alter-pk` is enabled in TiDB [#230](#) [#231](#)

14.7.19.3 New Features

- TiDB
 - Support sending Coprocessor requests to TiFlash in batches [#16226](#)
 - Enable the Coprocessor cache feature by default [#16710](#)

- Parse only the registered sections of a statement in the special comment of the SQL statement [#16157](#)
- Support using the `SHOW CONFIG` syntax to show the configurations of PD and TiKV instances [#16475](#)
- TiKV
 - Support using the user-owned KMS key for the server-side encryption when backing up data to S3 [#7630](#)
 - Enable the load-based `split region` operation [#7623](#)
 - Support validating common names [#7468](#)
 - Add the file lock check to avoid starting multiple TiKV instances that are bound to the same address [#7447](#)
 - Support AWS KMS in encryption at rest [#7465](#)
- Placement Driver (PD)
 - Remove `config manager` to let other components control their component configurations [#2349](#)
- TiFlash
 - Add the metrics report related to the read and write workloads of DeltaTree engine
 - Cache the `handle` and `version` columns to reduce the disk I/O of a single read or write request
 - Support pushing down the `fromUnixTime` and `dateFormat` functions
 - Evaluate the global state according to the first disk and report this evaluation
 - Add the graphics in Grafana related to the read and write workloads of DeltaTree engine
 - Optimize the decimal data encoding in the `Chunk` codec
 - Implement the gRPC API of Diagnostics (SQL diagnosis) to support querying system tables such as `INFORMATION_SCHEMA.CLUSTER_INFO`
- TiCDC
 - Support sending messages in batches in the Kafka sink module [#426](#)
 - Support file sorting in the processor [#477](#)
 - Support automatic `resolve lock` [#459](#)
 - Add the feature that automatically updates the TiCDC service GC safe point to PD [#487](#)
 - Add the timezone setting for data replication [#498](#)
- Backup and Restore (BR)
 - Support configuring S3/GCS in the storage URL [#246](#)

14.7.19.4 Bug Fixes

- TiDB
 - Fix the issue that negative numbers cannot be correctly displayed in the system table because the columns are defined as unsigned [#16004](#)
 - Add a warning when the `use_index_merge` hint contains the invalid index name [#15960](#)
 - Forbid multiple instances of a TiDB server sharing the same temporary directory [#16026](#)
 - Fix the panic that occurs during the execution of `explain for connection` when the plan cache is enabled [#16285](#)
 - Fix the issue that the result of the `tidb_capture_plan_baselines` system variable is incorrectly displayed [#16048](#)
 - Fix the issue that the `group by` clause in the `prepare` statement is incorrectly parsed [#16377](#)
 - Fix the panic that might occur during the execution of the `analyze primary key` statement [#16081](#)
 - Fix the issue that the TiFlash store information in the `cluster_info` system table is wrong [#16024](#)
 - Fix the panic that might occur during the Index Merge process [#16360](#)
 - Fix the issue that an incorrect result might occur when the Index Merge reader reads the generated columns [#16359](#)
 - Fix the incorrect display of the default sequence value in the `show create table` statement [#16526](#)
 - Fix the issue that the `not-null` error is returned because the sequence is used as the default values of the primary key [#16510](#)
 - Fix the issue that no error is reported for a blocked SQL execution when TiKV continues to return the `StaleCommand` error [#16530](#)
 - Fix the issue that an error is reported if you only specify `COLLATE` when creating a database; add the missing `COLLATE` part in the result of `SHOW CREATE DATABASE` [#16540](#)
 - Fix the partition pruning failure when the plan cache is enabled [#16723](#)
 - Fix the bug that `PointGet` returns wrong results when handling the overflow [#16755](#)

- Fix the issue that a wrong result is returned when querying the `slow_query` system table with equal time values [#16806](#)
- TiKV
 - Address the OpenSSL security issue: CVE-2020-1967 [#7622](#)
 - Avoid protecting rollback records written by `BatchRollback` to improve performance when many write conflicts exist in optimistic transactions [#7604](#)
 - Fix the issue that the needless wake-up of transactions results in useless retry and performance reduction in heavy lock-race workloads [#7551](#)
 - Fix the issue that the Region might be stuck in the multi-time merging [#7518](#)
 - Fix the issue that the learner is not deleted when deleting the learner [#7518](#)
 - Fix the issue that follower read might cause panic in raft-rs [#7408](#)
 - Fix the bug that a SQL operation might fail because of the `group by constant` error [#7383](#)
 - Fix the issue that an optimistic lock might block reads if the corresponding primary lock is a pessimistic lock [#7328](#)
- PD
 - Fix the issue that some APIs might fail in the TLS validation [#2363](#)
 - Fix the issue that the configuration API cannot accept a configuration item with a prefix [#2354](#)
 - Fix the issue that the 500 error is returned when the scheduler is not found [#2328](#)
 - Fix the issue that the 404 error is returned for the `scheduler config balance ↴ -hot-region-scheduler list` command [#2321](#)
- TiFlash
 - Disable the coarse-grained index optimization for the storage engine
 - Fix the bug that an exception is thrown when resolving locks for Regions and some locks need to be skipped
 - Fix the null pointer exception (NPE) when collecting the Coprocessor statistics
 - Fix the check for Region meta to ensure that the process of Region Split/Region Merge is correct
 - Fix the issue that the message size exceeds the limit for gRPC because the size of Coprocessor response is not estimated
 - Fix the handling of the `AdminCmdType::Split` command in TiFlash

14.7.20 TiDB 4.0 RC Release Notes

Release date: April 8, 2020

TiDB version: 4.0.0-rc

TiUP version: 0.0.3

Warning:

Some known issues are found in this version, and these issues are fixed in new versions. It is recommended that you use the latest 4.0.x version.

14.7.20.1 Compatibility Changes

- TiDB
 - Refuse to get started instead of returning an alert log when the tidb-server status port is occupied [#15177](#)
- TiKV
 - Support the `pipelined` feature in pessimistic transactions, which improves the TPC-C performance by 20%. The risk is that the transaction commit might fail because of lock failure during the execution [#6984](#)
 - Enable the `unify-read-pool` configuration item in new clusters by default and use the previous setting of this item in old clusters [#7059](#)
- Tools
 - TiDB Binlog
 - * Add the configuration item for verifying Common Name [#934](#)

14.7.20.2 Important Bug Fixes

- TiDB
 - Fix the issue that replication between the upstream and downstream might go wrong when the DDL job is executed using the `PREPARE` statement because of the incorrect job query in the internal records [#15435](#)
 - Fix the issue of incorrect subquery result in the `Read Committed` isolation level [#15471](#)
 - Fix the issue of incorrect results caused by the `Inline Projection` optimization [#15411](#)
 - Fix the issue that the SQL Hint `INL_MERGE_JOIN` is executed incorrectly in some cases [#15515](#)
 - Fix the issue that columns with the `AutoRandom` attribute are rebased when the negative number is explicitly written to these columns [#15397](#)

14.7.20.3 New Features

- TiDB
 - Add the case-sensitive collation so that users can enable `utf8mb4_general_ci` and `utf8_general_ci` in a new cluster [#33](#)
 - Enhance the `RECOVER TABLE` syntax to support recovering truncated tables [#15398](#)
 - Refuse to get started instead of returning an alert log when the the tidb-server status port is occupied [#15177](#)
 - Optimize the write performance of using a sequence as the default column values [#15216](#)
 - Add the `DDLJobs` system table to query the details of DDL jobs [#14837](#)
 - Optimize the `aggFuncSum` performance [#14887](#)
 - Optimize the output of `EXPLAIN` [#15507](#)
- TiKV
 - Support the `pipelined` feature in pessimistic transactions, which improves the TPC-C performance by 20%. The risk is that the transaction commit might fail because of lock failure during the execution [#6984](#)
 - Support TLS in the HTTP port [#5393](#)
 - Enable the `unify-read-pool` configuration item in new clusters by default and use the previous setting of this item in old clusters [#7059](#)
- PD
 - Support getting the default PD configuration information through the HTTP API [#2258](#)
- Tools
 - TiDB Binlog
 - * Add the configuration item for verifying Common Name [#934](#)
 - TiDB Lightning
 - * Optimize the performance of TiDB Lightning [#281](#) [#275](#)

14.7.20.4 Bug Fixes

- TiDB
 - Fix the issue that replication between the upstream and downstream might go wrong when the DDL job is executed using the `PREPARE` statement because of the incorrect job query in the internal records [#15435](#)

- Fix the issue of incorrect subquery result in the `Read Committed` isolation level [#15471](#)
- Fix the issue of possible wrong behavior when using `INSERT ... VALUES` to specify the `BIT(N)` data type [#15350](#)
- Fix the issue that the DDL Job internal retry does not fully achieve the expected outcomes because the values of `ErrorCount` fail to be summed correctly [#15373](#)
- Fix the issue that Garbage Collection might work abnormally when TiDB connects to TiFlash [#15505](#)
- Fix the issue of incorrect result caused by the Inline Projection optimization [#15411](#)
- Fix the issue that the SQL Hint `INL_MERGE_JOIN` is executed incorrectly in some cases [#15515](#)
- Fix the issue that columns with the `AutoRandom` attribute are rebased when the negative number is explicitly written to these columns [#15397](#)

- TiKV

- Fix the possible panic caused by transferring the leader when the Follower Read feature is enabled [#7101](#)

- Tools

- TiDB Lightning
 - * Fix the issue of data error caused by the error of character conversion when the backend is TiDB [#283](#)
 - TiCDC
 - * Fix the issue that an error is returned if the `test` schema does not exist in the downstream when MySQL sink is executing the DDL statement [#353](#)
 - * Support the real-time interactive mode in CDC cli [#351](#)
 - * Support checking whether the table in the upstream can be replicated during data replication [#368](#)
 - * Support asynchronous write to Kafka [#344](#)

14.7.21 TiDB 4.0.0 Beta.2 Release Notes

Release date: March 18, 2020

TiDB version: 4.0.0-beta.2

TiDB Ansible version: 4.0.0-beta.2

14.7.21.1 Compatibility Changes

- Tools

- TiDB Binlog
 - * Fix the issue that the system returns an error and exits when `disable-→ dispatch` and `disable-causality` are configured in Drainer [#915](#)

14.7.21.2 New Features

- TiKV
 - Support persisting the dynamically updated configuration into the hardware disk [#6684](#)
- PD
 - Support persisting the dynamically updated configuration into the hardware disk [#2153](#)
- Tools
 - TiDB Binlog
 - * Support the bidirectional data replication between TiDB clusters [#879](#) [#903](#)
 - TiDB Lightning
 - * Support the TLS configuration [#40](#) [#270](#)
 - TiCDC
 - * Initial release of the change data capture (CDC), providing the following features:
 - Support capturing changed data from TiKV
 - Support replicating the changed data from TiKV to MySQL compatible databases, and guarantee the eventual data consistency
 - Support replicating the changed data to Kafka, and guarantee either the eventual data consistency or the row-level orderliness
 - Provide process-level high availability
 - Backup & Restore (BR)
 - * Enable experimental features such as incremental backup and backing up files to Amazon S3 [#175](#)
- TiDB Ansible
 - Support injecting the node information to etcd [#1196](#)
 - Support deploying TiDB services on the ARM platform [#1204](#)

14.7.21.3 Bug Fixes

- TiKV
 - Fix the panic issue that might occur when meeting empty short values during the backup [#6718](#)

- Fix the issue that Hibernate Regions might not be correctly awakened in some cases [#6772](#) [#6648](#) [#6376](#)
- PD
 - Fix the panic issue that the rule checker fails to allocate stores to Regions [#2160](#)
 - Fix the issue that after the dynamic configuration is enabled, the configuration might have replication delay when the Leader is being switched [#2154](#)
- Tools
 - Backup & Restore (BR)
 - * Fix the issue that BR might fail to restore data of a large size because PD cannot process large-sized data [#167](#)
 - * Fix the BR failure occurred because the BR version is not compatible with the TiDB version [#186](#)
 - * Fix the BR failure occurred because the BR version is not compatible with TiFlash [#194](#)

14.7.22 TiDB 4.0.0 Beta.1 Release Notes

Release date: February 28, 2020

TiDB version: 4.0.0-beta.1

TiDB Ansible version: 4.0.0-beta.1

14.7.22.1 Compatibility Changes

- TiDB
 - Modify the type of the `log.enable-slow-log` configuration item from integer to Boolean [#14864](#)
 - Modify the `password` field name to `authentication_string` in the `mysql.user` → system table to make it consistent with MySQL 5.7 (**This compatibility change means that you cannot roll back to earlier versions.**) [#14598](#)
 - Adjust the default value of the `txn-total-size-limit` configuration item from 1GB to 100MB [#14522](#)
 - Support dynamically modifying or updating configuration items read from PD [#14750](#) [#14303](#) [#14830](#)
- TiKV
 - Add the `readpool.unify-read-pool` configuration item (True by default) to control whether point queries use the same threads with Coprocessor [#6375](#) [#6401](#) [#6534](#) [#6582](#) [#6585](#) [#6593](#) [#6597](#) [#6677](#)
- PD

- Optimize the HTTP API to make it compatible with the configuration manager [#2080](#)
- TiDB Lightning
 - Use the default configurations specified in the document for certain items not configured in the configuration file [#255](#)
- TiDB Ansible
 - Rename `theflash` to `tiflash` [#1130](#)
 - Optimize the default values and related configurations in TiFlash's configuration file [#1138](#)

14.7.22.2 New Features

- TiDB
 - Support querying slow logs of any time in the `SLOW_QUERY` / `CLUSTER_SLOW_QUERY` → system table [#14840](#) [#14878](#)
 - Support SQL performance diagnosis
 - * [#14843](#) [#14810](#) [#14835](#) [#14801](#) [#14743](#)
 - * [#14718](#) [#14721](#) [#14670](#) [#14663](#) [#14668](#)
 - * [#14896](#)
 - Support the `Sequence` function [#14731](#) [#14589](#) [#14674](#) [#14442](#) [#14303](#) [#14830](#)
 - Support dynamically modifying or updating configuration items read from PD [#14750](#) [#14303](#) [#14830](#)
 - Add a feature of automatically reading data from different roles according to the load balancing policy and add the `leader-and-follower` system variable to enable this feature [#14761](#)
 - Add the `Coercibility` function [#14739](#)
 - Support setting TiFlash replicas in the partitioned table [#14735](#) [#14713](#) [#14644](#)
 - Improve the privilege check for the `SLOW_QUERY` table [#14451](#)
 - Support automatically write the intermediate results to the disk file if the memory is insufficient when using a SQL join [#14708](#) [#14279](#)
 - Support checking table partitions by querying the `information_schema`. → `PARTITIONS` system table [#14347](#)
 - Add the `json_objectagg` aggregate function [#11154](#)
 - Support logging rejected connection attempts in the audit log [#14594](#)
 - Add the `max-server-connections` configuration item (4096 by default) to control the number of connections to a single server [#14409](#)
 - Support the isolation read specifying multiple storage engines at the server level [#14440](#)
 - Optimize the cost model of the `Apply` operator and the `Sort` operator to improve stability [#13550](#) [#14708](#)
- TiKV

- Support fetching configuration items from the status port via HTTP API [#6480](#)
- Optimize the performance of `Chunk Encoder` in Coprocessor [#6341](#)
- PD
 - Support accessing to the distribution of hotspots in the cluster through Dashboard UI [#2086](#)
 - Support capturing and displaying `START_TIME` and `UPTIME` of cluster components [#2116](#)
 - Add the information of deployment path and component version in the returned message of the `member` API [#2130](#)
 - Add the `component` sub-command in `pd-ctl` to modify and check the configuration of other components (experimental) [#2092](#)
- TiDB Binlog
 - Support TLS between the components [#904](#) [#894](#)
 - Add the `kafka-client-id` configuration item in Drainer to configure Kafka's client ID [#902](#)
 - Support purging the incremental backup data in Drainer [#885](#)
- TiDB Ansible
 - Support deploying multiple Grafana/Prometheus/Alertmanagers in one cluster [#1142](#)
 - Add the `metric_port` configuration item (8234 by default) in TiFlash's configuration file [#1145](#)
 - Add the `flash_proxy_status_port` configuration item (20292 by default) in TiFlash's configuration file [#1141](#)
 - Add the TiFlash monitoring dashboard [#1147](#) [#1151](#)

14.7.22.3 Bug Fixes

- TiDB
 - Fix the issue that an error is reported when creating `view` with a column name that exceeds 64 characters [#14850](#)
 - Fix the issue that duplicate data exists in `information_schema.views` because the `create or replace view` statement is incorrectly processed [#14832](#)
 - Fix the incorrect results of `BatchPointGet` when `plan cache` is enabled [#14855](#)
 - Fix the issue that data is inserted into the wrong partitioned table after the timezone is modified [#14370](#)
 - Fix the panic occurred when rebuilding expression using the invalid name of the `IsTrue` function during the outer join simplification [#14515](#)
 - Fix the the incorrect privilege check for the `show binding` statement [#14443](#)
- TiKV

- Fix the inconsistent behaviors of the `CAST` function in TiDB and TiKV [#6463](#) [#6461](#) [#6459](#) [#6474](#) [#6492](#) [#6569](#)
- TiDB Lightning
 - Fix the bug that the web interface does not work outside the Server mode [#259](#)

14.7.23 TiDB 4.0 Beta Release Notes

Release date: January 17, 2020

TiDB version: 4.0.0-beta

TiDB Ansible version: 4.0.0-beta

14.7.23.1 TiDB

- Print the log or cancel the SQL execution when the memory used during the execution of `INSERT/REPLACE/DELETE/UPDATE` exceeds the limit specified by the `MemQuotaQuery` configuration item. The actual behavior depends on the `OOMAction` configuration. [#14179](#) [#14289](#) [#14299](#)
- Increase the accuracy of calculating the cost of `Index Join` by considering the row counts of both driving tables and driven tables [#12085](#)
- Add 15 SQL hints to control the behavior of the optimizer and make the optimizer more stable
 - [#11253](#) [#11364](#) [#11673](#) [#11740](#) [#11746](#)
 - [#11809](#) [#11996](#) [#12043](#) [#12059](#) [#12246](#)
 - [#12382](#)
- Improve the performance when the columns involved in a query can be fully covered by indexes [#12022](#)
- Improve the performance of table query by supporting the Index Merge feature [#10121](#) [#10512](#) [#11245](#) [#12225](#) [#12248](#) [#12305](#) [#12843](#)
- Improve the performance of Range calculation and reduce the CPU overhead by caching index results and eliminating duplicate results [#12856](#)
- Decouple the level of slow logs from the level of ordinary logs [#12359](#)
- Add the `oom-use-tmp-storage` parameter (`true` by default) to control whether to use temporary files to cache intermediate results when the memory usage for the execution of a single SQL statement exceeds `mem-quota-query` and the SQL contains `Hash Join` [#11832](#) [#11937](#) [#12116](#) [#12067](#)
- Support using `create index/alter table` to create expression index and using `drop index` to drop expression index [#14117](#)
- Increase the default value of the `query-log-max-len` parameter to 4096 to reduce the number of truncated SQL outputs. This parameter can be adjusted dynamically. [#12491](#)

- Support adding the `AutoRandom` keyword in the column attribute to control whether the system automatically assigns a random integer to the primary key, which avoids the hotspot problem caused by the `AUTO_INCREMENT` primary key [#13127](#)
- Support Table Locks [#11038](#)
- Support using the `LIKE` or `WHERE` clause in `ADMIN SHOW DDL JOBS` for conditional filtering [#12484](#)
- Add the `TIDB_ROW_ID_SHARDING_INFO` column in the `information_schema.tables` table to output the RowID scattering information (for example, the value of the `SHARD_ROW_ID_BITS` column in table A is "SHARD_BITS={bit_number}") [#13418](#)
- Optimize the error code of SQL error messages to avoid the situation that the `ERROR 1105 (HY000)` code is used for multiple error messages (the `Unknown Error` type)
 - [#14002](#) [#13874](#) [#13733](#) [#13654](#) [#13646](#)
 - [#13540](#) [#13366](#) [#13329](#) [#13300](#) [#13233](#)
 - [#13033](#) [#12866](#) [#14054](#)
- Convert a narrow data range of the discrete type into `point set` and use CM-Sketch to improve the estimation accuracy when estimating the number of rows [#11524](#)
- Extract the TopN information from CM-Sketch for normal `Analyze` and separately maintain the frequently occurring values [#11409](#)
- Support dynamically adjusting the depth and width of CM-Sketch and the number of TopN information [#11278](#)
- Support automatically capturing and evolving SQL Binding [#13199](#) [#12434](#)
- Optimize the encoding format of communication with TiKV by using `Chunk` to improve communication performance [#12023](#) [#12536](#) [#12613](#) [#12621](#) [#12899](#) [#13060](#) [#13349](#)
- Support the new row store format to improve the performance of the wide table [#12634](#)
- Optimize the `Recover Binlog` interface to ensure waiting all transactions to be committed before returning to the client [#13740](#)
- Support querying the binlog statuses enabled by TiDB servers in the cluster through the HTTP `info/all` interface [#13025](#)
- Support the MySQL-compatible `Read Committed` transaction isolation level when using the pessimistic transaction mode [#14087](#)
- Support large-sized transactions. The transaction size is limited by the size of the physical memory.
 - [#11999](#) [#11986](#) [#11974](#) [#11817](#) [#11807](#)
 - [#12133](#) [#12223](#) [#12980](#) [#13123](#) [#13299](#)
 - [#13432](#) [#13599](#)
- Improve the stability of `Kill` [#10841](#)
- Support hexadecimal and binary expressions as separators in `LOAD DATA` [#11029](#)
- Improve the performance of `IndexLookupJoin` and reduce memory consumption during execution by splitting `IndexLookupJoin` into `IndexHashJoin` and `IndexMergeJoin` [#8861](#) [#12139](#) [#12349](#) [#13238](#) [#13451](#) [#13714](#)
- Fix several issues relating to RBAC [#13896](#) [#13820](#) [#13940](#) [#14090](#) [#13940](#) [#13014](#)
- Fix the issue that `VIEW` cannot be created because the `SELECT` statement contains `union` [#12595](#)

- Fix several issues relating to the `CAST` function
 - #12858 #11968 #11640 #11483 #11493
 - #11376 #11355 #11114 #14405 #14323
 - #13837 #13401 #13334 #12652 #12864
 - #12623 #11989
- Output the detailed `backoff` information of TiKV RPC in the slow log to facilitate troubleshooting [#13770](#)
- Optimize and unify the format of the memory statistics in the expensive log [#12809](#)
- Optimize the explicit format of `EXPLAIN` and support outputting information about the operator's usage of memory and disk [#13914](#) [#13692](#) [#13686](#) [#11415](#) [#13927](#) [#13764](#) [#13720](#)
- Optimize the check for duplicate values in `LOAD DATA` based on the transaction size and support setting the transaction size by configuring the `tidb_dml_batch_size` parameter [#11132](#)
- Optimize the performance of `LOAD DATA` by separating the data preparing routine and the commit routine and assigning the workload to different Workers [#11533](#) [#11284](#)

14.7.23.2 TiKV

- Upgrade the RocksDB version to 6.4.6
- Fix the issue that the system cannot perform the compaction task normally when the disk space is used up by automatically creating a 2GB empty file when TiKV is started [#6321](#)
- Support quick backup and restoration
 - #6462 #6395 #6378 #6374 #6349
 - #6339 #6308 #6295 #6286 #6283
 - #6261 #6222 #6209 #6204 #6202
 - #6198 #6186 #6177 #6146 #6071
 - #6042 #5877 #5806 #5803 #5800
 - #5781 #5772 #5689 #5683
- Support reading data from Follower replicas
 - #5051 #5118 #5213 #5316 #5401
 - #5919 #5887 #6340 #6348 #6396
- Improve the performance of TiDB reading data through index [#5682](#)
- Fix the issue that the `CAST` function behaves inconsistently in TiKV and in TiDB
 - #6459 #6461 #6458 #6447 #6440
 - #6425 #6424 #6390 #5842 #5528
 - #5334 #5199 #5167 #5146 #5141
 - #4998 #5029 #5099 #5006 #5095
 - #5093 #5090 #4987 #5066 #5038

- #4962 #4890 #4727 #6060 #5761
- #5793 #5468 #5540 #5548 #5455
- #5543 #5433 #5431 #5423 #5179
- #5134 #4685 #4650 #6463

14.7.23.3 PD

- Support optimizing hotspot scheduling according to the load information of storage nodes
 - #1870 #1982 #1998 #1843 #1750
- Add the Placement Rules feature that supports controlling the number of replicas of any data range, the storage location, the storage host type and roles by combining different scheduling rules
 - #2051 #1999 #2042 #1917 #1904
 - #1897 #1894 #1865 #1855 #1834
- Support using plugins (experimental) #1799
- Add the feature that the schedulers support the customized configuration and key ranges (experimental) #1735 #1783 #1791
- Support automatically adjusting the scheduling speed according the cluster load information (experimental, disabled by default) #1875 #1887 #1902

14.7.23.4 Tools

- TiDB Lightning
 - Add the parameter in the command-line tool to set the password of the downstream database #253

14.7.23.5 TiDB Ansible

- Add checksum check in the package in case that the downloaded package is incomplete #1002
- Support checking the systemd version which must be `systemd-219-52` or later #1020 #1074
- Fix the issue that the log directory is incorrectly created when TiDB Lightning is started #1103
- Fix the issue that the customized port of TiDB Lightning is invalid #1107
- Support deploying and maintaining TiFlash #1119

14.8 v3.1

14.8.1 TiDB 3.1.2 Release Notes

Release date: June 4, 2020

TiDB version: 3.1.2

14.8.1.1 Bug Fixes

- TiKV
 - Fix the error handling issue during backup and restoration with S3 and GCS [#7965](#)
 - Fix the `DefaultNotFound` error that occurs during restoration [#7838](#)
- Tools
 - Backup & Restore (BR)
 - * Retry automatically when the network is poor to improve stability with S3 and GCS storages [#314](#) [#7965](#)
 - * Fix a restoration failure that occurs because the Region leader cannot be found when restoring small tables [#303](#)
 - * Fix a data loss issue during restoration when a table's row ID exceeds 2^{63} [#323](#)
 - * Fix the issue that empty databases and tables cannot be restored [#318](#)
 - * Support using AWS KMS for server-side encryption (SSE) when targeting the S3 storage [#261](#)

14.8.2 TiDB 3.1.1 Release Notes

Release date: April 30, 2020

TiDB version: 3.1.1

TiDB Ansible version: 3.1.1

14.8.2.1 New Features

- TiDB
 - Add the table option for `auto_rand_base` [#16812](#)
 - Add the `Feature_ID` comment: In the special comments of SQL statements, only the registered statement fragment can be parsed by the parser; otherwise, the statement is ignored [#16155](#)

- TiFlash
 - Cache the `handle` and `version` columns to reduce the disk I/O for a single read request
 - Add in Grafana the graphics related to the read and write workloads of DeltaTree engine
 - Optimize the decimal data encoding in the `Chunk` codec
 - Reduce the number of open file descriptors when TiFlash is in low workload

14.8.2.2 Bug Fixes

- TiDB
 - Fix the issue that the isolation read setting at the instance level does not take effect, and that the isolation read setting is incorrectly retained after TiDB is upgraded [#16482](#) [#16802](#)
 - Fix the partition selection syntax on the hash partitioned table so that an error is not reported for syntaxes such as `partition (P0)` [#16076](#)
 - Fix the issue that when an UPDATE SQL statement only queries from a view but does not update the view, the update statement still reports an error [#16789](#)
 - Fix the issue of wrong results caused by removing the `not` `not` from the nested query [#16423](#)
- TiFlash
 - Fix the issue that an error occurs when reading data from a Region that is in the abnormal state
 - Modify the mapping of table names in TiFlash to correctly support `recover ↳ table/flashback table`
 - Modify the storage path to fix the potential data loss issue that occurs when renaming a table
 - Modify the read mode in the online update scenario to improve the read performance
 - Fix the issue that TiFlash fails to start normally after upgrade if the database/table name contains special characters
- Tools
 - Backup & Restore (BR)
 - * Fix the issue that after BR restores a table with the `auto_random` attribute, inserting data might trigger the duplicate entry error [#241](#)

14.8.3 TiDB 3.1.0 GA Release Notes

Release date: April 16, 2020

TiDB version: 3.1.0 GA

TiDB Ansible version: 3.1.0 GA

14.8.3.1 Compatibility Changes

- TiDB
 - Support directly stopping starting TiDB if the HTTP listening port is unavailable when the `report-status` configuration item is enabled [#16291](#)
- Tools
 - Backup & Restore (BR)
 - * BR does not support restoring data from the TiKV cluster earlier than 3.1 GA [#233](#)

14.8.3.2 New Features

- TiDB
 - Support displaying the information of Coprocessor tasks in `explain format = ↳ "dot"` [#16125](#)
 - Reduce the redundant stack information of log using the `disable-error-stack` configuration item [#16182](#)
- Placement Driver (PD)
 - Optimize the hot Region scheduling [#2342](#)
- TiFlash
 - Add the metrics report related to the read and write workloads of DeltaTree engine
 - Support pushing down the `fromUnixTime` and `dateFormat` functions
 - Disable the rough set filter by default
- TiDB Ansible
 - Add TiFlash monitor [#1253](#) [#1257](#)
 - Optimize the configuration parameters of TiFlash [#1262](#) [#1265](#) [#1271](#)
 - Optimize the TiDB starting script [#1268](#)

14.8.3.3 Bug Fixes

- TiDB
 - Fix the panic issue caused by the merge join operation in some scenarios [#15920](#)
 - Fix the issue that some expressions are repeatedly counted in selectivity calculation [#16052](#)
 - Fix the panic issue occurred when loading the statistics information in extreme cases [#15710](#)
 - Fix the issue that an error is returned in some cases when equivalent expressions cannot be recognized in SQL query [#16015](#)
 - Fix the issue that an error is returned when querying the `view` of one database from another database [#15867](#)
 - Fix the panic issue that occurs when the column is handled using `fast analyze` [#16080](#)
 - Fix the incorrect character set of the `current_role` print result [#16084](#)
 - Refine the log of MySQL connection handshake error [#15799](#)
 - Fix the panic issue caused by port probing after the audit plugin is loaded [#16065](#)
 - Fix the panic issue of the `sort` operator on left join because the `TypeNull` class is mistaken as a variable-length type [#15739](#)
 - Fix the issue of inaccurate count of monitoring session retry errors [#16120](#)
 - Fix the issue of wrong results of `weekday` in the `ALLOW_INVALID_DATES` mode [#16171](#)
 - Fix the issue that Garbage Collection (GC) might not work normally when the cluster has TiFlash nodes [#15761](#)
 - Fix the issue that TiDB goes out of memory (OOM) when users set a large partition count when creating the hash partitioned table [#16219](#)
 - Fix the issue that warnings are mistaken as errors, and make the `UNION` statement have the same behavior as the `SELECT` statement [#16138](#)
 - Fix the execution error when TopN is pushed down to mocktikv [#16200](#)
 - Increase the initial length of `chunk.column.nullBitMap` to avoid unnecessary overhead of `runtime.growslice` [#16142](#)
- TiKV
 - Fix the panic issue caused by replica read [#7418](#) [#7369](#)
 - Fix the issue that the restoration process creates empty Regions [#7419](#)
 - Fix the issue that repeated resolve lock requests might harm the atomicity of pessimistic transactions [#7389](#)
- TiFlash
 - Fix the potential issue of the `rename table` operation when replicating the schema from TiDB
 - Fix the issue of data loss caused by the `rename table` operation under multiple data path configurations

- Fix the issue that TiFlash reports incorrect storage space in some scenarios
 - Fix the potential issue caused by reading from TiFlash when Region Merge is enabled
- Tools
 - TiDB Binlog
 - * Fix the issue that TiFlash-related DDL jobs might interrupt the replication of Drainer [#948](#) [#942](#)
 - Backup & Restore (BR)
 - * Fix the issue that the `checksum` operation is still executed when it is disabled [#223](#)
 - * Fix the issue that incremental backup fails when TiDB enables `auto-random` or `alter-pk` [#230](#) [#231](#)

14.8.4 TiDB 3.1 RC Release Notes

Release date: April 2, 2020

TiDB version: 3.1.0-rc

TiDB Ansible version: 3.1.0-rc

Warning:

Some known issues are found in this version, and these issues are fixed in new versions. It is recommended that you use the latest 3.1.x version.

14.8.4.1 New Features

- TiDB
 - Use the binary search to re-implement partition pruning for better performance [#15678](#)
 - Support using the `RECOVER` syntax to recover the truncated table [#15460](#)
 - Add the `AUTO_RANDOM` ID cache for retrying statements and recovering tables [#15393](#)
 - Support restoring the state of the `AUTO_RANDOM` ID allocator using the `recover ↳ table` statement [#15393](#)
 - Support `YEAR`, `MONTH`, and `TO_DAY` functions as the partitioning keys of the Hash partitioned table [#15619](#)

- Add the table ID to the schema-change related tables only when keys need to be locked in the `SELECT... FOR UPDATE` statement [#15708](#)
- Add the feature of automatically reading data from different roles according to the load balancing policy and add the `leader-and-follower` system variable to enable this feature [#15721](#)
- Support dynamically updating the TLS certificate every time TiDB establishes a new connection to update expired client certificate without restarting the RPC client side [#15163](#)
- Upgrade PD Client to support loading the latest certificate every time TiDB establishes a new connection [#15425](#)
- Forcibly use the HTTPS protocol with the configured TLS certificates between a TiDB server and a PD server, or between two TiDB servers when `cluster-ssl-*` is configured [#15430](#)
- Add the MySQL-compatible `--require-secure-transport` startup option to force the client to enable TLS authentication during the configuration [#15442](#)
- Add the `cluster-verify-cn` configuration item. After configuration, the status service can only be used when with the corresponding CN certificate [#15137](#)

- TiKV

- Support backing up data with the Raw KV API [#7051](#)
- Support TLS authentication for the status server [#7142](#)
- Support TLS authentication for the KV server [#7305](#)
- Optimize the time to hold locks to improve the performance of backup [#7202](#)

- PD

- Support scheduling learner using `shuffle-region-scheduler` [#2235](#)
- Add commands in pd-ctl to configure Placement Rules [#2306](#)

- Tools

- TiDB Binlog
 - * Support TLS authentication between the components [#931](#) [#937](#) [#939](#)
 - * Add the `kafka-client-id` configuration item in Drainer to configure Kafka's client ID [#929](#)
- TiDB Lightning
 - * Optimize the performance of TiDB Lightning [#281](#) [#275](#)
 - * Support TLS authentication for TiDB Lightning [#270](#)
- Backup & Restore (BR)
 - * Optimize the log output [#189](#)

- TiDB Ansible

- Optimize the way the TiFlash data directories are created [#1242](#)
- Add the `Write Amplification` monitoring item in TiFlash [#1234](#)
- Optimize the error message of failed preflight checks when CPU epollexclusive is unavailable [#1243](#)

14.8.4.2 Bug Fixes

- TiDB

- Fix the information schema error caused by frequently updating the TiFlash replica [#14884](#)
- Fix the issue that `last_insert_id` is incorrectly generated when applying `AUTO_RANDOM` [#15149](#)
- Fix the issue that updating the status of TiFlash replica might cause the DDL operation to get stuck [#15161](#)
- Forbid Aggregation pushdown and TopN pushdown when there are predicates that can not be pushed down [#15141](#)
- Forbid the nested `view` creation [#15440](#)
- Fix the error occurred when executing `SELECT CURRENT_ROLE()` after `SET ROLE ↪ ALL` [#15570](#)
- Fix the failure to identify the `view` name when executing the `select view_name ↪ .col_name from view_name` statement [#15573](#)
- Fix the issue that an error might occur when pre-processing DDL statements during the write of binlog information [#15444](#)
- Fix the panic occurred when accessing both `views` and partitioned tables [#15560](#)
- Fix the error occurred when executing the `VALUES` function with the `update ↪ duplicate key` statement that contains the `bit(n)` data type [#15487](#)
- Fix the issue that the specified maximum execution time fails to take effect in some scenarios [#15616](#)
- Fix the issue that whether the current `ReadEngine` contains TiKV server is not checked when generating the execution plan using `Index Scan` [#15773](#)

- TiKV

- Fix the issue of conflict check failure or data index inconsistency caused by inserting an existing key into a transaction and then deleting it immediately when disabling the consistency check parameter [#7112](#)
- Fix the calculation error when TopN compares unsigned integers [#7199](#)
- Introduce a flow control mechanism in Raftstore to solve the problem that without flow control, it might cause slow log tracking and cause the cluster to be stuck; and the problem that the large transaction size might cause the frequent reconnection among TiKV servers [#7087](#) [#7078](#)
- Fix the issue that pending read requests sent to replicas might be permanently blocked [#6543](#)
- Fix the issue that replica read might be blocked by applying snapshots [#7249](#)

- Fix the issue that transferring leader might cause TiKV to panic [#7240](#)
- Fix the issue that all SST files are filled with zeroes when backing up data to S3 [#6967](#)
- Fix the issue that the size of SST file is not recorded during backup, resulting in many empty Regions after restoration [#6983](#)
- Support AWS IAM web identity for backup [#7297](#)
- PD
 - Fix the issue of incorrect Region information caused by data race when PD processes Region heartbeats [#2234](#)
 - Fix the issue that `random-merge-scheduler` fails to follow location labels and Placement Rules [#2212](#)
 - Fix the issue that a placement rule is overwritten by another placement rule with the same `startKey` and `endKey` [#2222](#)
 - Fix the issue that the version number of API is inconsistent with that of PD server [#2192](#)
- Tools
 - TiDB Lightning
 - * Fix the bug that the & character is replaced by the EOF character in TiDB backend [#283](#)
 - Backup & Restore (BR)
 - * Fix the issue that BR cannot restore the TiFlash cluster data [#194](#)

14.8.5 TiDB 3.1 Beta.2 Release Notes

Release date: March 9, 2020

TiDB version: 3.1.0-beta.2

TiDB Ansible version: 3.1.0-beta.2

Warning:

Some known issues are found in this version, and these issues are fixed in new versions. It is recommended that you use the latest 3.1.x version.

14.8.5.1 Compatibility Changes

- Tools
 - TiDB Lightning
 - * Use the default configurations specified in the [TiDB Lightning Configuration](#) for certain items not configured in the configuration file [#255](#)
 - * Add the `--tidb-password` CLI parameter to set the TiDB password [#253](#)

14.8.5.2 New Features

- TiDB
 - Support adding the `AutoRandom` keyword in the column attribute to enable TiDB to automatically assign random integers to the primary key, which avoids the write hot spot caused by the `AUTO_INCREMENT` primary key [#14555](#)
 - Support creating or deleting column store replicas through DDL statements [#14537](#)
 - Add the feature that the optimizer can independently select different storage engines [#14537](#)
 - Add the feature that the SQL hint supports different storage engines [#14537](#)
 - Support reading data from followers by using the `tidb_replica_read` system variable [#13464](#)
- TiKV
 - Raftstore
 - * Add the `peer_address` parameter to connect other nodes to the TiKV server [#6491](#)
 - * Add the `read_index` and `read_index_resp` monitoring metrics to monitor the number of ReadIndex requests [#6610](#)
- PD Client
 - Support reporting statistics of local threads to PD [#6605](#)
- Backup
 - Replace the `RocksIOLimiter` flow control library with Rust's `async-speed-limit` ↵ flow control library to eliminate extra memory copies when backing up a file [#6462](#)
- PD
 - Tolerate backslash in the location label name [#2084](#)
- TiFlash
 - Initial release

- TiDB Ansible
 - Support deploying multiple Grafana/Prometheus/Alertmanager in one cluster [#1143](#)
 - Support deploying the TiFlash component [#1148](#)
 - Add monitoring metrics related to the TiFlash component [#1152](#)

14.8.5.3 Bug Fixes

- TiKV
 - Raftstore
 - * Fix the issue that the read requests cannot be processed because data is not properly read from Hibernate Regions [#6450](#)
 - * Fix the panic issue caused by the `ReadIndex` requests during the leader transfer process [#6613](#)
 - * Fix the issue that Hibernate Regions are not correctly awakened in some special conditions [#6730](#) [#6737](#) [#6972](#)
 - Backup
 - * Fix the inconsistent data index during the restoration caused by the backup of the extra data [#6659](#)
 - * Fix the panic caused by incorrectly processing the deleted values during the backup [#6726](#)
- PD
 - Fix the panic occurred because the rule checker fails to assign stores to Regions [#2161](#)
- Tools
 - TiDB Lightning
 - * Fix the bug that the web interface does not work outside the Server mode [#259](#)
 - BR (Backup and Restore)
 - * Fix the issue that BR cannot exit in time due to an unrecoverable error it encounters when restoring data [#152](#)
- TiDB Ansible
 - Fix the issue that the rolling update command fails because the PD Leader cannot be obtained in some scenarios [#1122](#)

14.8.6 TiDB 3.1 Beta.1 Release Notes

Release date: January 10, 2020

TiDB version: 3.1.0-beta.1

TiDB Ansible version: 3.1.0-beta.1

14.8.6.1 TiKV

- backup
 - Change the name of the backup file from `start_key` to the hash value of `start_key` to reduce the file name's length for easy reading [#6198](#)
 - Disable RocksDB's `force_consistency_checks` check to avoid false positives in the consistency check [#6249](#)
 - Add the incremental backup feature [#6286](#)
- sst_importer
 - Fix the issue that the SST file does not have MVCC properties during restoring [#6378](#)
 - Add the monitoring items such as `tikv_import_download_duration`, `tikv_import_download_latency`, `tikv_import_ingest_duration`, `tikv_import_ingest_bytes`, and `tikv_import_error_counter` to observe the overheads of downloading and ingesting SST files [#6404](#)
- raftstore
 - Fix the issue of Follower Read that the follower reads stale data when the leader changes, thus breaking transaction isolation [#6343](#)

14.8.6.2 Tools

- BR (Backup and Restore)
 - Fix the inaccurate backup progress information [#127](#)
 - Improve the performance of splitting Regions [#122](#)
 - Add the backup and restore feature for partitioned tables [#137](#)
 - Add the feature of automatically scheduling PD schedulers [#123](#)
 - Fix the issue that data is overwritten after non PKIsHandle tables are restored [#139](#)

14.8.6.3 TiDB Ansible

- Add the feature of automatically disabling Transparent Huge Pages (THP) in the operating system during the initialization phase [#1086](#)
- Add the Grafana monitoring for BR components [#1093](#)
- Optimize the deployment of TiDB Lightning by automatically creating related directories [#1104](#)

14.8.7 TiDB 3.1 Beta Release Notes

Release date: December 20, 2019

TiDB version: 3.1.0-beta

TiDB Ansible version: 3.1.0-beta

14.8.7.1 TiDB

- SQL Optimizer
 - Enrich SQL hints [#12192](#)
- New feature
 - Support the Follower Read feature [#12535](#)

14.8.7.2 TiKV

- Support the distributed backup and restore feature [#5532](#)
- Support the Follower Read feature [#5562](#)

14.8.7.3 PD

- Support the distributed backup and restore feature [#1896](#)

14.9 v3.0

14.9.1 TiDB 3.0.20 Release Notes

Release date: December 25, 2020

TiDB version: 3.0.20

14.9.1.1 Compatibility Change

- TiDB
 - Deprecate the `enable-streaming` configuration item [#21054](#)

14.9.1.2 Improvements

- TiDB
 - Raise an error when preparing the `LOAD DATA` statement [#21222](#)
- TiKV
 - Add the `end_point_slow_log_threshold` configuration item [#9145](#)

14.9.1.3 Bug Fixes

- TiDB
 - Fix the incorrect cache of the transaction status for pessimistic transactions [#21706](#)
 - Fix the issue of inaccurate statistics that occurs when querying `INFORMATION_SCHEMA` → `.TIDB_HOT_REGIONS` [#21319](#)
 - Fix the issue that `DELETE` might not delete data correctly when the database name is not in a pure lower representation [#21205](#)
 - Fix the issue of stack overflow that occurs when building the recursive view [#21000](#)
 - Fix the issue of goroutine leak in TiKV client [#20863](#)
 - Fix the wrong default zero value for the `year` type [#20828](#)
 - Fix the issue of goroutine leak in index lookup join [#20791](#)
 - Fix the issue that executing `INSERT SELECT FOR UPDATE` returns the malformed packet in the pessimistic transaction [#20681](#)
 - Fix the unknown time zone '`posixrules`' [#20605](#)
 - Fix the issue that occurs when converting the unsigned integer type to the bit type [#20362](#)
 - Fix the corrupted default value of the bit type column [#20339](#)
 - Fix the potentially incorrect results when one of the equal condition is the `Enum` or `Set` type [#20296](#)
 - Fix a wrong behavior of `!= any()` [#20061](#)
 - Fix the issue that type conversion in `BETWEEN...AND...` returns invalid results [#21503](#)
 - Fix a compatibility issue with the `ADDDATE` function [#21008](#)
 - Set the correct default value for newly added `Enum` column [#20999](#)
 - Fix the result of SQL statements like `SELECT DATE_ADD('2007-03-28` → `22:08:28', INTERVAL "-2.-2" SECOND)` to be compatible with MySQL [#20627](#)
 - Fix the incorrect default value when modifying the column type [#20532](#)
 - Fix the issue that the `timestamp` function gets wrong result when the input argument is the `float` or `decimal` type [#20469](#)
 - Fix a potential deadlock issue in statistics [#20424](#)
 - Fix the issue that the overflowed float type data is inserted [#20251](#)
- TiKV
 - Fix the issue that an error is returned indicating that a key exists when this key is locked and deleted in a committed transaction [#8931](#)
- PD
 - Fix the issue that too many logs are printed when starting PD and when there are too many stale Regions [#3064](#)

14.9.2 TiDB 3.0.19 Release Notes

Release date: September 25, 2020

TiDB version: 3.0.19

14.9.2.1 Compatibility Changes

- PD
 - Change the import path from pingcap/pd to tikv/pd [#2779](#)
 - Change the copyright information from PingCAP, Inc to TiKV Project Authors [#2777](#)

14.9.2.2 Improvements

- TiDB
 - Mitigate the impact of failure recovery on QPS performance [#19764](#)
 - Support adjusting the concurrency of the `union` operator [#19885](#)
- TiKV
 - Set `sync-log` to `true` as a nonadjustable value [#8636](#)
- PD
 - Add an alert rule for PD restart [#2789](#)

14.9.2.3 Bug Fixes

- TiDB
 - Fix the query error that occurs when the `slow-log` file does not exist [#20050](#)
 - Add the privilege check for `SHOW STATS_META` and `SHOW STATS_BUCKET` [#19759](#)
 - Forbid changing the decimal type to the integer type [#19681](#)
 - Fix the issue that the constraint is not checked when altering the `ENUM/SET` type column [#20045](#)
 - Fix the bug that tidb-server does not release table locks after a panic [#20021](#)
 - Fix the bug that the `OR` operator is not handled correctly in the `WHERE` clause [#19901](#)
- TiKV

- Fix the bug that TiKV panics when parsing responses with missing reason phrases [#8540](#)
- Tools
 - TiDB Lightning
 - * Fix the issue that the TiDB Lightning process does not exit in time when encountering illegal UTF characters in CSV in the strict mode [#378](#)

14.9.3 TiDB 3.0.18 Release Notes

Release date: August 21, 2020

TiDB version: 3.0.18

14.9.3.1 Improvements

- Tools
 - TiDB Binlog
 - * Support the time duration format of Go for the Pump GC configuration [#996](#)

14.9.3.2 Bug Fixes

- TiDB
 - Fix the issue that the wrong handling of the `decimal` type by the `Hash` function causes the wrong HashJoin result [#19185](#)
 - Fix the issue that the wrong handling of the `set` and `enum` types by the `Hash` function causes the wrong HashJoin result [#19175](#)
 - Fix the issue that the check for duplicate keys fails in the pessimistic locking mode [#19236](#)
 - Fix the issue that the `Apply` and `Union Scan` operators cause the wrong execution result [#19297](#)
 - Fix the issue that some cached execution plans are incorrectly executed in transaction [#19274](#)
- TiKV
 - Change the GC failure log from `error` to the `warning` level [#8444](#)
- Tools
 - TiDB Lightning
 - * Fix the issue that the `--log-file` argument does not take effect [#345](#)
 - * Fix the syntax error on empty binary/hex literals when using TiDB-backend [#357](#)
 - * Fix the unexpected `switch-mode` call when using TiDB-backend [#368](#)

14.9.4 TiDB 3.0.17 Release Notes

Release date: Aug 3, 2020

TiDB version: 3.0.17

14.9.4.1 Improvements

- TiDB

- Decrease the default value of the `query-feedback-limit` configuration item from 1024 to 512, and improve the statistics feedback mechanism to ease its impact on the cluster [#18770](#)
- Limit batch split count for one request [#18694](#)
- Accelerate `/tiflash/replica` HTTP API when there are many history DDL jobs in the TiDB cluster [#18386](#)
- Improve row count estimation for index equal condition [#17609](#)
- Speed up the execution of `kill tidb conn_id` [#18506](#)

- TiKV

- Add the `hibernate-timeout` configuration that delays region hibernation to improve rolling update performance [#8207](#)

- Tools

- TiDB Lightning
 - * `[black-white-list]` has been deprecated with a newer, easier-to-understand filter format [#332](#)

14.9.4.2 Bug Fixes

- TiDB

- Return the actual error message instead of an empty set when a query which contains `IndexHashJoin` or `IndexMergeJoin` encounters a panic [#18498](#)
- Fix the unknown column error for SQL statements like `SELECT a FROM t HAVING ↪ t.a` [#18432](#)
- Forbid adding a primary key for a table when the table has no primary key or when the table already has an integer primary key [#18342](#)
- Return an empty set when executing `EXPLAIN FORMAT="dot" FOR CONNECTION #17157`
- Fix `STR_TO_DATE`'s handling for format token '`%r`', '`%h`' [#18725](#)

- TiKV

- Fix a bug that might read stale data during region merging [#8111](#)
- Fix the issue of memory leak during the scheduling process [#8355](#)
- Tools
 - TiDB Lightning
 - * Fix the issue that the `log-file` flag is ignored [#345](#)

14.9.5 TiDB 3.0.16 Release Notes

Release date: July 03, 2020

TiDB version: 3.0.16

14.9.5.1 Improvements

- TiDB
 - Support the `is null` filter condition in hash partition pruning [#17308](#)
 - Assign different `Backoffers` to each Region to avoid the SQL timeout issue when multiple Region requests fail at the same time [#17583](#)
 - Split separate Regions for the newly added partition [#17668](#)
 - Discard feedbacks generated from the `delete` or `update` statement [#17841](#)
 - Correct the usage of `json.Unmarshal` in `job.DecodeArgs` to be compatible with future Go versions [#17887](#)
 - Remove sensitive information in the slow query log and the statement summary table [#18128](#)
 - Match the MySQL behavior with `DateTime` delimiters [#17499](#)
 - Handle `%h` in date formats in the range that is consistent with MySQL [#17496](#)
- TiKV
 - Avoid sending store heartbeats to PD after snapshots are received [#8145](#)
 - Improve the PD client log [#8091](#)

14.9.5.2 Bug Fixes

- TiDB
 - Fix the data inconsistency issue occurred because the lock of a written and deleted primary key in one transaction is resolved by another transaction [#18248](#)
 - Fix the `Got too many pings` gRPC error log in the PD server-side followers [17944](#)
 - Fix the panic issue that might occur when the child of HashJoin returns the `TypeNull` column [#17935](#)

- Fix the error message when access is denied [#17722](#)
 - Fix JSON comparison issue for the `int` and `float` types [#17715](#)
 - Update the failpoint which causes data race [#17710](#)
 - Fix the issue that the timeout pre-split Regions might not work when creating tables [#17617](#)
 - Fix the panic caused by ambiguous error messages after the sending failure [#17378](#)
 - Fix the issue that `FLASHBACK TABLE` might fail in some special cases [#17165](#)
 - Fix the issue of inaccurate range calculation results when statements only have string columns [#16658](#)
 - Fix the query error occurred when the `only_full_group_by` SQL mode is set [#16620](#)
 - Fix the issue that the field length of results returned from the `case when` function is inaccurate [#16562](#)
 - Fix the type inference for the decimal property in the `count` aggregate function [#17702](#)
- TiKV
 - Fix the potential wrong result read from ingested files [#8039](#)
 - Fix the issue that a peer can not be removed when its store is isolated during multiple merge processes [#8005](#)
 - PD
 - Fix the 404 error when querying Region keys in PD Control [#2577](#)

14.9.6 TiDB 3.0.15 Release Notes

Release date: June 5, 2020

TiDB version: 3.0.15

14.9.6.1 New Features

- TiDB
 - Forbid the query in partitioned tables to use the plan cache feature [#16759](#)
 - Support the `admin recover index` and `admin check index` statements on partitioned tables [#17315](#) [#17390](#)
 - Support partition pruning of the `in` condition for Range partitioned tables [#17318](#)
 - Optimize the output of `SHOW CREATE TABLE`, and add quotation marks to the partition name [#16315](#)
 - Support the `ORDER BY` clause in the `GROUP_CONCAT` function [#16988](#)
 - Optimize the memory allocation mechanism of `CMSketch` statistics to reduce the impact of garbage collection (GC) on performance [#17543](#)

- PD
 - Add a policy in which PD performs scheduling in terms of the number of Leaders [#2479](#)

14.9.6.2 Bug Fixes

- TiDB
 - Use deep copy to copy the `enum` and `set` type data in the `Hash` aggregate function; fix an issue of correctness [#16890](#)
 - Fix the issue that `PointGet` returns incorrect results because of the wrong processing logic of integer overflow [#16753](#)
 - Fix the issue of incorrect results caused by incorrect processing logic when the `CHAR()` function is used in the query predicate [#16557](#)
 - Fix the issue of inconsistent results in the storage layer and calculation layer of the `IsTrue` and `IsFalse` functions [#16627](#)
 - Fix the incorrect `NotNull` flags in some expressions, such as `case when` [#16993](#)
 - Fix the issue that the optimizer cannot find a physical plan for `TableDual` in some scenarios [#17014](#)
 - Fix the issue that the syntax for partition selection does not take effect correctly in the Hash partitioned table [#17051](#)
 - Fix the inconsistent results between TiDB and MySQL when XOR operates on a floating-point number [#16976](#)
 - Fix the error that occurs when executing DDL statement in the prepared manner [#17415](#)
 - Fix the incorrect processing logic of computing the batch size in the ID allocator [#17548](#)
 - Fix the issue that the `MAX_EXEC_TIME` SQL hint does not take effect when the time exceeds the expensive threshold [#17534](#)
- TiKV
 - Fix the issue that memory defragmentation is not effective after running for a long time [#7790](#)
 - Fix the panic issue caused by incorrectly removing snapshot files after TiKV is restarted accidentally [#7925](#)
 - Fix the gRPC disconnection caused by too large message packages [#7822](#)

14.9.7 TiDB 3.0.14 Release Notes

Release date: May 9, 2020

TiDB version: 3.0.14

14.9.7.1 Compatibility Changes

- TiDB
 - Adjust the user privilege in `performance_schema` and `metrics_schema` from read-write to read-only [#15417](#)

14.9.7.2 Important Bug Fixes

- TiDB
 - Fix the issue that the query result of `index join` is incorrect when the `join ↪` condition has multiple equivalent conditions on the column with the `handle` attribute [#15734](#)
 - Fix the panic that occurs when performing the `fast analyze` operation on the column with the `handle` attribute [#16079](#)
 - Fix the issue that the `query` field in the DDL job structure is incorrect when the DDL statement is executed in a way of `prepare`. This issue might cause data inconsistency between the upstream and the downstream when Binlog is used for data replication. [#15443](#)
- TiKV
 - Fix the issue that repeated requests on the cleanup of lock might destroy the atomicity of the transaction [#7388](#)

14.9.7.3 New Features

- TiDB
 - Add the schema name column and the table name column to the query results of the `admin show ddl jobs` statement [#16428](#)
 - Enhance the `RECOVER TABLE` syntax to support recovering truncated tables [#15458](#)
 - Support the privilege check for the `SHOW GRANTS` statement [#16168](#)
 - Support the privilege check for the `LOAD DATA` statement [#16736](#)
 - Improve the performance of partition pruning when functions related to time and date are used as partition keys [#15618](#)
 - Adjust the log level of `dispatch error` from `WARN` to `ERROR` [#16232](#)
 - Support the `require-secure-transport` startup option to force clients to use TLS [#15415](#)
 - Support HTTP communication between TiDB components when TLS is configured [#15419](#)

- Add the `start_ts` information of the current transaction to the `information_schema` → `.processlist` table [#16160](#)
- Support automatically reloading the TLS certificate information used for communication among clusters [#15162](#)
- Improve the read performance of the partitioned tables by restructuring the partition pruning [#15628](#)
- Support the partition pruning feature when `floor(unix_timestamp(a))` is used as the partition expression of the `range` partition table [#16521](#)
- Allow executing the `update` statement that contains a `view` and does not update the `view` [#16787](#)
- Prohibit creating nested `views` [#15424](#)
- Prohibit truncating `view` [#16420](#)
- Prohibit using the `update` statement to explicitly update the values of a column when this column is not in the `public` state [#15576](#)
- Prohibit starting TiDB when the `status` port is occupied [#15466](#)
- Change the character set of the `current_role` function from `binary` to `utf8mb4` [#16083](#)
- Improve `max-execution-time` usability by checking the interrupt signal when the data of a new Region is read [#15615](#)
- Add the `ALTER TABLE ... AUTO_ID_CACHE` syntax for explicitly setting the cache step of `auto_id` [#16287](#)
- TiKV
 - Improve the performance when many conflicts and the `BatchRollback` condition exist in optimistic transactions [#7605](#)
 - Fix the issue of decreased performance that occurs because the pessimistic lock `waiter` is frequently awakened when many conflicts exist in pessimistic transactions [#7584](#)
- Tools
 - TiDB Lightning
 - * Support printing the TiKV cluster mode using the `fetch-mode` sub-command of `tidb-lightning-ctl` [#287](#)

14.9.7.4 Bug Fixes

- TiDB
 - Fix the issue that `WEEKEND` function is not compatible with MySQL when the SQL mode is `ALLOW_INVALID_DATES` [#16170](#)
 - Fix the issue that the `DROP INDEX` statement fails to execute when the index column contains the auto-increment primary key [#16008](#)

- Fix the issue of incorrect values of the `TABLE_NAMES` column in the Statement Summary [#15231](#)
- Fix the issue that some expressions have incorrect results when the plan cache is enabled [#16184](#)
- Fix the issue that the result of the `not/istrue/isfalse` function is incorrect [#15916](#)
- Fix the panic caused by the `MergeJoin` operation on tables with redundant indexes [#15919](#)
- Fix the issue caused by incorrectly simplifying the link when the predicate only refers to the outer table [#16492](#)
- Fix the issue that the `CURRENT_ROLE` function reports an error caused by the `SET ROLE` statement [#15569](#)
- Fix the issue that the result of the `LOAD DATA` statement is incompatible with MySQL when this statement encounters \ [#16633](#)
- Fix the issue that the database visibility is incompatible with MySQL [#14939](#)
- Fix the issue of incorrect privilege check for the `SET DEFAULT ROLE ALL` statement [#15585](#)
- Fix the issue of partition pruning failure caused by the plan cache [#15818](#)
- Fix the issue that `schema change` is reported during the transaction commit when concurrent DDL operations are performed on a table and blocking exists, because the transaction does not lock the related table [#15707](#)
- Fix the incorrect behavior of `IF(not_int, *, *)` [#15356](#)
- Fix the incorrect behavior of `CASE WHEN (not_int)` [#15359](#)
- Fix the issue that the `Unknown column` error message is returned when using a `view` that is not in the current schema [#15866](#)
- Fix the issue that the result of parsing time strings is incompatible with MySQL [#16242](#)
- Fix the possible panic of the collation operator in `left join` when a `null` column exists in the right child node [#16528](#)
- Fix the issue that no error message is returned even though the SQL execution is blocked when TiKV keeps returning the `StaleCommand` error message [#16528](#)
- Fix the possible panic caused by the port probing when the audit plugin is enabled [#15967](#)
- Fix the panic caused when `fast analyze` works on indices only [#15967](#)
- Fix the possible panic of the `SELECT * FROM INFORMATION_SCHEMA.PROCESSLIST` statement execution in some cases [#16309](#)
- Fix the issue of TiDB OOM caused by specifying a large number of partitions (for example, 999999999999) when the hash partition table is created without checking the number of partitions before allocating memory [#16218](#)
- Fix the issue of incorrect information of partitioned tables in `information_schema` $\hookrightarrow .tidb_hot_table$ [#16726](#)
- Fix the issue that the partition selection algorithm does not take effect on the hash partitioned table [#16070](#)
- Fix the issue that the HTTP API of the MVCC series does not support partitioned tables [#16191](#)

- Keep the error handling of the UNION statement consistent with that of the SELECT statement [#16137](#)
 - Fix the issue of incorrect behavior when the parameter type of the VALUES function is `bit(n)` [#15486](#)
 - Fix the issue that the processing logic of TiDB is inconsistent with MySQL when the `view` column name is too long. In this case, the system automatically generates a short column name. [#14873](#)
 - Fix the issue that `(not not col)` is incorrectly optimized as `col` [#16094](#)
 - Fix the issue of incorrect `range` of the inner table built by `IndexLookupJoin` plans [#15753](#)
 - Fix the issue that `only_full_group_by` fails to correctly check expressions with brackets [#16012](#)
 - Fix the issue that an error is returned when the `select view_name.col_name` → `from view_name` statement is executed [#15572](#)
- TiKV
 - Fix the issue that the node cannot be deleted correctly after the isolation recovery in some cases [#7703](#)
 - Fix the issue of data loss during network isolation caused by the Region Merge operation [#7679](#)
 - Fix the issue that learner cannot be removed correctly in some cases [#7598](#)
 - Fix the issue that the scanning result of raw key-value pairs might be out of order [#7597](#)
 - Fix the issue of reconnection when the batch of Raft messages is too large [#7542](#)
 - Fix the issue of gRPC thread deadlock caused by the empty request [#7538](#)
 - Fix the issue that the processing logic of restarting the learner is incorrect during the merge process [#7457](#)
 - Fix the issue that repeated requests on the cleanup of lock might destroy the atomicity of the transaction [#7388](#)

14.9.8 TiDB 3.0.13 Release Notes

Release date: April 22, 2020

TiDB version: 3.0.13

14.9.8.1 Bug Fixes

- TiDB
 - Fix the issue caused by unchecked `MemBuffer` that the `INSERT ... ON` → `DUPLICATE KEY UPDATE` statement might be executed incorrectly within a transaction when users need to insert multiple rows of duplicate data [#16690](#)

- TiKV
 - Fix the issue that the system might get stuck and the service is unavailable if Region Merge is executed repeatedly [#7612](#)

14.9.9 TiDB 3.0.12 Release Notes

Release date: March 16, 2020

TiDB version: 3.0.12

TiDB Ansible version: 3.0.12

Warning:

Some known issues are found in this version, and these issues are fixed in new versions. It is recommended that you use the latest 3.0.x version.

14.9.9.1 Compatibility Changes

- TiDB
 - Fix the issue of inaccurate timing of prewrite binlog in slow query log. The original timing field was called `Binlog_prewrite_time`. After this fix, the name is changed to `Wait_prewrite_binlog_time`. [#15276](#)

14.9.9.2 New Features

- TiDB
 - Support dynamic loading of the replaced certificate file by using the `alter ↳ instance` statement [#15080](#) [#15292](#)
 - Add the `cluster-verify-cn` configuration item. After configuration, the status service can only be used when with the corresponding CN certificate. [#15164](#)
 - Add a flow limiting feature for DDL requests in each TiDB server to reduce the error reporting frequency of DDL request conflicts [#15148](#)
 - Support exiting of the TiDB server when binlog write fails [#15339](#)
- Tools
 - TiDB Binlog
 - * Add the `kafka-client-id` configuration item in Drainer, which supports connecting to Kafka clients to configure the client ID [#929](#)

14.9.9.3 Bug Fixes

- TiDB
 - Make GRANT, REVOKE guarantee atomicity when modifying multiple users [#15092](#)
 - Fix the issue that the locking of pessimistic lock on the partition table failed to lock the correct row [#15114](#)
 - Make the error message display according to the value of max-index-length in the configuration when the index length exceeds the limit [#15130](#)
 - Fix the incorrect decimal point issue of the FROM_UNIXTIME function [#15270](#)
 - Fix the issue of conflict detection failure or data index inconsistency caused by deleting records written by oneself in a transaction [#15176](#)
- TiKV
 - Fix the issue of conflict detection failure or data index inconsistency caused by inserting an existing key into a transaction and then deleting it immediately when disabling the consistency check parameter [#7054](#)
 - Introduce a flow control mechanism in Raftstore to solve the problem that without flow control, it might lead to too slow tracking and cause the cluster to be stuck, and the transaction size might cause frequent reconnection of TiKV connections [#7072](#) [#6993](#)
- PD
 - Fix the issue of incorrect Region information caused by data race when PD processes Region heartbeats [#2233](#)
- TiDB Ansible
 - Support deploying multiple Grafana/Prometheus/Alertmanager in a cluster [#1198](#)

14.9.10 TiDB 3.0.11 Release Notes

Release date: March 4, 2020

TiDB version: 3.0.11

TiDB Ansible version: 3.0.11

Warning:

Some known issues are found in this version, and these issues are fixed in new versions. It is recommended that you use the latest 3.0.x version.

14.9.10.1 Compatibility Changes

- TiDB
 - Add the `max-index-length` configuration item to control the maximum index length, which is compatible with the behavior of TiDB versions before 3.0.7 or of MySQL [#15057](#)

14.9.10.2 New Features

- TiDB
 - Support showing the meta information of partitioned tables in the `information_schema` ↵ `.PARTITIONS` table [#14849](#)
- TiDB Binlog
 - Support the bidirectional data replication between TiDB clusters [#884](#) [#909](#)
- TiDB Lightning
 - Support the TLS configuration [#44](#) [#270](#)
- TiDB Ansible
 - Modify the logic of `create_users.yml` so that users of the control machine do not have to be consistent with `ansible_user` [#1184](#)

14.9.10.3 Bug Fixes

- TiDB
 - Fix the issue of Goroutine leaks when retrying an optimistic transaction because queries using `Union` are not marked read-only [#15076](#)
 - Fix the issue that `SHOW TABLE STATUS` fails to correctly output the table status at the snapshot time because the value of the `tidb_snapshot` parameter is not correctly used when executing the `SET SESSION tidb_snapshot = 'xxx'`; statement [#14391](#)
 - Fix the incorrect result caused by a SQL statement that contains `Sort Merge` ↵ `Join` and `ORDER BY DESC` at the same time [#14664](#)
 - Fix the panic of TiDB server when creating partition tables using the unsupported expression. The error information `This partition function is not allowed` is returned after fixing this panic. [#14769](#)
 - Fix the incorrect result occurred when executing the `select max() from` ↵ `subquery` statement with the subquery containing `Union` [#14944](#)
 - Fix the issue that an error message is returned when executing the `SHOW BINDINGS` statement after executing `DROP BINDING` that drops the execution binding [#14865](#)

- Fix the issue that the connection is broken because the maximum length of an alias in a query is 256 characters in the MySQL protocol, but TiDB does not [cut the alias](#) in the query results according to this protocol [#14940](#)
- Fix the incorrect query result that might occur when using the string type in DIV. For instance, now you can correctly execute the `select 1 / '2007' div 1` statement [#14098](#)
- TiKV
 - Optimize the log output by removing unnecessary logs [#6657](#)
 - Fix the panic that might occur when the peer is removed under high loads [#6704](#)
 - Fix the issue that Hibernate Regions are not waken up in some cases [#6732](#) [#6738](#)
- TiDB Ansible
 - Update outdated document links in `tidb-ansible` [#1169](#)
 - Fix the issue that undefined variables might occur in the `wait for region ↳ replication complete` task [#1173](#)

14.9.11 TiDB 3.0.10 Release Notes

Release date: February 20, 2020

TiDB version: 3.0.10

TiDB Ansible version: 3.0.10

Warning:

Some known issues are found in this version, and these issues are fixed in new versions. It is recommended that you use the latest 3.0.x version.

14.9.11.1 TiDB

- Fix wrong Join results when `IndexLookUpJoin` uses `OtherCondition` to construct `InnerRange` [#14599](#)
- Delete the `tidb_pprof_sql_cpu` configuration item and add the `tidb_pprof_sql_cpu` variable [#14416](#)
- Fix the issue that users can query all databases only when they have global privileges [#14386](#)
- Fix the issue that data visibility does not meet expectations due to transaction timeout when executing `PointGet` operations [#14480](#)
- Change the timing of pessimistic transaction activation to delayed activation, consistent with the optimistic transaction mode [#14474](#)

- Fix the incorrect time zone results when the `unixtimestamp` expression calculates the time zone of the table partitions [#14476](#)
- Add the `tidb_session_statement_deadlock_detect_duration_seconds` monitoring item to monitor deadlock detection duration [#14484](#)
- Fix the system panic issue caused by some logic errors of GC workers [#14439](#)
- Correct the expression name of the `IsTrue` function [#14516](#)
- Fix the issue that some memory usage is counted inaccurately [#14533](#)
- Fix the system panic issue caused by incorrect processing logic during CM-Sketch statistics initialization [#14470](#)
- Fix the issue of inaccurate partition pruning when querying partitioned tables [#14546](#)
- Fix the issue that the default database name of the SQL statement in SQL bindings is set incorrectly [#14548](#)
- Fix the issue that `json_key` is not compatible with MySQL [#14561](#)
- Add the feature of automatically updating the statistics of partitioned tables [#14566](#)
- Fix the issue that the plan ID changes when the `PointGet` operation is executed (the plan ID is expected to be 1 always) [#14595](#)
- Fix the system panic issue caused by incorrect processing logic when SQL bindings do not match exactly [#14263](#)
- Add the `tidb_session_statement_pessimistic_retry_count` monitoring item to monitor the number of retries after the failure to lock pessimistic transactions [#14619](#)
- Fix the incorrect privilege check for `show binding` statements [#14618](#)
- Fix the issue that the query cannot be killed because the `backoff` logic does not include checking the `killed` tag [#14614](#)
- Improve the performance of statement summary by reducing the time to hold internal locks [#14627](#)
- Fix the issue that TiDB's result of parsing strings to time is incompatible with MySQL [#14570](#)
- Record the user login failures in audit logs [#14620](#)
- Add the `tidb_session_statement_lock_keys_count` monitoring item to monitor the number of lock keys for pessimistic transactions [#14634](#)
- Fix the issue that characters in JSON such as <, <, and > are incorrectly escaped [#14637](#)
- Fix the system panic issue caused by excessive memory usage when the `HashJoin` operation is building a hash table [#14642](#)
- Fix the panic issue caused by incorrect processing logic when an SQL binding processes illegal records [#14645](#)
- Fix a MySQL incompatibility issue by adding Truncated error detection to decimal division calculation [#14673](#)
- Fix the issue of successfully granting users privileges on a table that does not exist [#14611](#)

14.9.11.2 TiKV

- Raftstore

- Fix the system panic issue #6460 or data loss issue #598 caused by Region merge failure [#6481](#)
- Support `yield` to optimize scheduling fairness, and support pre-transferring the leader to improve leader scheduling stability [#6563](#)

14.9.11.3 PD

- Fix the invalid cache issue by supporting automatically updating the Region cache information when the system traffic changes [#2103](#)
- Use leader lease time to determine TSO service validity [#2117](#)

14.9.11.4 Tools

- TiDB Binlog
 - Support relay log in Drainer [#893](#)
- TiDB Lightning
 - Make some configuration items use default values when a config file is missing [#255](#)
 - Fix the issue that the web interface cannot be opened in the non-server mode [#259](#)

14.9.11.5 TiDB Ansible

- Fix the issue that the command execution fails due to the failure to obtain PD leader in some scenarios [#1121](#)
- Add the `Deadlock Detect Duration` monitoring item in the TiDB dashboard [#1127](#)
- Add the `Statement Lock Keys Count` monitoring item in the TiDB dashboard [#1132](#)
- Add the `Statement Pessimistic Retry Count` monitoring item in the TiDB dashboard [#1133](#)

14.9.12 TiDB 3.0.9 Release Notes

Release date: January 14, 2020

TiDB version: 3.0.9

TiDB Ansible version: 3.0.9

Warning:

Some known issues are found in this version, and these issues are fixed in new versions. It is recommended that you use the latest 3.0.x version.

14.9.12.1 TiDB

- Executor
 - Fix the incorrect result when the aggregate function is applied to the ENUM column and the collection column [#14364](#)
- Server
 - Support the `auto_increment_increment` and `auto_increment_offset` system variables [#14396](#)
 - Add the `tidb_tikvclient_ttl_lifetime_reach_total` monitoring metric to monitor the number of pessimistic transactions with a TTL of 10 minutes [#14300](#)
 - Output the SQL information in the log when the SQL query causes a panic during its execution [#14322](#)
 - Add the `plan` and `plan_digest` fields in the statement summary table to record the `plan` that is being executed and the `plan` signature [#14285](#)
 - Adjust the default value of the `stmt-summary.max-stmt-count` configuration item from 100 to 200 [#14285](#)
 - Add the `plan_digest` field in the slow query table to record the `plan` signature [#14292](#)
- DDL
 - Fix the issue that the results of anonymous indexes created using `alter table ↪ ... add index` on the `primary` column is inconsistent with MySQL [#14310](#)
 - Fix the issue that VIEWS are mistakenly dropped by the `drop table` syntax [#14052](#)
- Planner
 - Optimize the performance of statements such as `select max(a), min(a) ↪ from t`. If an index exists in the `a` column, the statement is optimized to `select * from (select a from t order by a desc limit 1)as t1, ↪ (select a from t order by a limit 1)as t2` to avoid full table scan [#14410](#)

14.9.12.2 TiKV

- Raftstore
 - Speed up the configuration change to speed up the Region scattering [#6421](#)
- Transaction
 - Add the `tikv_lock_manager_waiter_lifetime_duration`, `tikv_lock_manager_detect_duration`, and `tikv_lock_manager_detect_duration` monitoring metrics to monitor waiters' lifetime, the time cost of detecting deadlocks, and the status of `Wait` table [#6392](#)

- Optimize the following configuration items to reduce transaction execution latency caused by changing Region leader or the leader of deadlock detector in extreme situations [#6429](#)
 - * Change the default value of `wait-for-lock-time` from `3s` to `1s`
 - * Change the default value of `wake-up-delay-duration` from `100ms` to `20ms`
- Fix the issue that the leader of the deadlock detector might be incorrect during the Region Merge process [#6431](#)

14.9.12.3 PD

- Support using backlash / in the location label name [#2083](#)
- Fix the incorrect statistics because the tombstone store is mistakenly included by the label counter [#2067](#)

14.9.12.4 Tools

- TiDB Binlog
 - Add the unique key information in the binlog protocol output by Drainer [#862](#)
 - Support using the encrypted password for database connection for Drainer [#868](#)

14.9.12.5 TiDB Ansible

- Support automatically creating directories to optimize the deployment of TiDB Lightning [#1105](#)

14.9.13 TiDB 3.0.8 Release Notes

Release date: December 31, 2019

TiDB version: 3.0.8

TiDB Ansible version: 3.0.8

14.9.13.1 TiDB

- SQL Optimizer
 - Fix the wrong SQL binding plan caused by untimely cache updates [#13891](#)
 - Fix the issue that the SQL binding might be invalid when an SQL statement contains a symbol list [#14004](#)
 - Fix the issue that an SQL binding cannot be created or deleted because an SQL statement ends with ; [#14113](#)

- Fix the issue that a wrong SQL query plan might be selected because the `PhysicalUnionScan` operator sets wrong statistics [#14133](#)
- Remove the `minAutoAnalyzeRatio` restriction to make `autoAnalyze` more timely [#14015](#)
- SQL Execution Engine
 - Fix issues that the `INSERT/REPLACE/UPDATE ... SET ... = DEFAULT` syntax might report an error and combining the usage of the `DEFAULT` expression with a virtual generated column might report an error [#13682](#)
 - Fix the issue that the `INSERT` statement might report an error when converting a string to a float [#14011](#)
 - Fix the issue that sometimes the aggregate operation is low effective because the concurrency value of the `HashAgg` executor is incorrectly initialized [#13811](#)
 - Fix the issue that an error is reported in the execution of `group by item` when the clause is in the parentheses [#13658](#)
 - Fix the issue that the execution of `OUTER JOIN` might report an error because TiDB incorrectly calculates `group by item` [#14014](#)
 - Fix the issue that the error message is inaccurate when Range-exceeding data is written into Range partitioned tables [#14107](#)
 - Revert PR [#10124](#) and cancel the `PadCharToFullLength` effect to avoid unexpected query results in special cases, considering that MySQL 8 will discard `PadCharToFullLength` soon [#14157](#)
 - Fix the goroutine leak issue when executing the `EXPLAIN ANALYZE` statement caused by unguaranteed `close()` calling in `ExplainExec` [#14226](#)
- DDL
 - Optimize the error message output of `change column/modify column` to make it easier to understand [#13796](#)
 - Add the `SPLIT PARTITION TABLE` syntax to support splitting Regions for partitioned tables [#13929](#)
 - Fix the issue that the index length exceeds 3072 bytes and no error is reported because the index length is incorrectly checked when an index is created [#13779](#)
 - Fix the issue that the `GC life time is shorter than transaction duration` error message might be reported because it takes too much time to add an index in partitioned tables [#14132](#)
 - Fix the panic when `SELECT * FROM information_schema.KEY_COLUMN_USAGE` is executed because the foreign key is not checked when `DROP COLUMN/MODIFY ↗ COLUMN/CHANGE COLUMN` is executed [#14105](#)
- Server
 - Statement Summary improvements:
 - * Add a large number of SQL metric fields to facilitate analyzing SQL statements in more detail [#14151](#), [#14168](#)

- * Add the `stmt-summary.refresh-interval` parameter to control whether to move the stale data from the `events_statements_summary_by_digest` table to the `events_statements_summary_by_digest_history` table (the default interval: 30 minutes) [#14161](#)
- * Add the `events_statements_summary_by_digest_history` table to save the stale data in `events_statements_summary_by_digest` [#14166](#)
- Fix the issue that the binlog is incorrectly output when RBAC-related internal SQL statements are executed [#13890](#)
- Add the `server-version` configuration item to control the feature of modifying the TiDB server version [#13906](#)
- Add the feature of using the HTTP interface to recover writing the TiDB binlog [#13892](#)
- Update the privilege required by `GRANT roles TO user` from `GrantPriv` to `ROLE_ADMIN` or `SUPER`, to keep consistency with the MySQL behavior [#13932](#)
- Modify the TiDB behavior from using the current database to reporting the `No → database selected` error when the `GRANT` statement does not specify a database name, to keep compatibility with the MySQL behavior [#13784](#)
- Modify the execution privilege for the `REVOKE` statement from `SuperPriv` to `REVOKE` being executable only if the user has the privilege for the corresponding schema, to keep consistency with the MySQL behavior [#13306](#)
- Fix the issue that `GrantPriv` is mistakenly granted to the target user when the `GRANT ALL` syntax does not contain `WITH GRANT OPTION` [#13943](#)
- Fix the issue that the error message does not contain the cause for the `LOAD DATA` statement's wrong behavior when `LoadDataInfo` fails to call `addRecord` [#13980](#)
- Fix the issue that wrong slow query information is output because multiple SQL statements in a query share the same `StartTime` [#13898](#)
- Fix the issue that the memory might leak when `batchClient` processes a large transaction [#14032](#)
- Fix the issue that `system_time_zone` is always displayed as CST and now TiDB's `system_time_zone` is obtained from `systemTZ` in the `mysql.tidb` table [#14086](#)
- Fix the issue that the `GRANT ALL` syntax does not grant all privileges to the user [#14092](#)
- Fix the issue that the `Priv_create_user` privilege is invalid for `CREATE ROLE` and `DROP ROLE` [#14088](#)
- Modify the error code of `ErrInvalidFieldSize` from `1105(Unknow Error)` to `3013` [#13737](#)
- Add the `SHUTDOWN` command to stop a TiDB server and add the `ShutdownPriv` privilege [#14104](#)
- Fix the atomicity issue for the `DROP ROLE` statement to avoid some roles being deleted unexpectedly when TiDB fails to execute a statement [#14130](#)
- Fix the issue that the `tidb_enable_window_function` in the `SHOW VARIABLE` result incorrectly outputs 1 when a TiDB version is upgraded to 3.0, and replace the wrong result with 0 [#14131](#)
- Fix the issue that the goroutine might leak because `gcworker` continuously retries

when the TiKV node is offline [#14106](#)

- Record the binlog Prewrite time in the slow query log to improve the usability for issue tracking [#14138](#)
- Make the `tidb_enable_table_partition` variable support GLOBAL SCOPE [#14091](#)
- Fix the issue that the user privilege might be missing or mistakenly added because the newly added privilege is not correctly granted to the corresponding user when a new privilege is added [#14178](#)
- Fix the issue that the `CheckStreamTimeoutLoop` goroutine might leak because `rpcClient` does not close when the TiKV server is disconnected [#14227](#)
- Support certificate-based authentication ([User document](#)) [#13955](#)
- Transaction
 - Update the default value of the `tidb_txn_mode` variable from "" to "pessimistic" ↳ " when a new cluster is created [#14171](#)
 - Fix the issue that the lock waiting time is too long for a pessimistic transaction because the lock waiting time for a single statement is not reset when a transaction is retried [#13990](#)
 - Fix the issue that wrong data might be read because unmodified data is unlocked for the pessimistic transaction mode [#14050](#)
 - Fix repeated insert value restriction checks because transaction types are not distinguished when prewrite is performed in mocktikv [#14175](#)
 - Fix the panic because transactions are not correctly handled when `session.` ↳ `TxnState is Invalid` [#13988](#)
 - Fix the issue that the `ErrConfclit` structure in mocktikv does not contain `ConflictCommitTS` [#14080](#)
 - Fix the issue that the transaction is blocked because TiDB does not correctly check lock timeout after resolving the lock [#14083](#)
- Monitor
 - Add the `pessimistic_lock_keys_duration` monitoring item in `LockKeys` [#14194](#)

14.9.13.2 TiKV

- Coprocessor
 - Modify the level of the output log from `error` to `warn` when an error occurs in Coprocessor [#6051](#)
 - Modify the update behavior of statistics sampling data from directly updating the row to deleting before inserting, to keep consistency with the update behavior of tidb-server [#6069](#)
- Raftstore

- Fix the panic caused by repeatedly sending the `destroy` message to `peer fsm` and `peer fsm` being destroyed multiple times [#6297](#)
- Update the default value of `split-region-on-table` from `true` to `false` to disable splitting Regions by table by default [#6253](#)
- Engine
 - Fix the issue that empty data might be returned because RocksDB iterator errors are not correctly processed in extreme conditions [#6326](#)
- Transaction
 - Fix the issue that TiKV fails to write data into keys and GC is blocked because the pessimistic locks are incorrectly cleaned up [#6354](#)
 - Optimize the pessimistic lock waiting mechanism to improve the performance in scenarios where the lock conflict is severe [#6296](#)
- Update the default value of `tikv_alloc` from `tikv_alloc/default` to `jemalloc` [#6206](#)

14.9.13.3 PD

- Client
 - Support using `context` to create a client and setting the timeout duration when creating a new client [#1994](#)
 - Support creating the `KeepAlive` connection [#2035](#)
- Optimize the performance for the `/api/v1/regions` API [#1986](#)
- Fix the issue that deleting stores in a `tombstone` state might cause a panic [#2038](#)
- Fix the issue that overlapped Regions are mistakenly deleted when loading the Region information from disks [#2011](#), [#2040](#)
- Upgrade etcd from v3.4.0 to v3.4.3 (note that after upgrading you can only degrade etcd using pd-recover) [#2058](#)

14.9.13.4 Tools

- TiDB Binlog
 - Fix the issue that the binlog is ignored because Pump does not receive the DDL committed binlog [#853](#)

14.9.13.5 TiDB Ansible

- Revert the simplified configuration item [#1053](#)
- Optimize the logic for checking the TiDB version when performing a rolling update [#1056](#)

- Upgrade TiSpark to v2.1.8 [#1061](#)
- Fix the issue that the PD role monitoring item is wrongly displayed on Grafana [#1065](#)
- Optimize Thread Voluntary Context Switches and Thread Nonvoluntary → Context Switches monitoring items on the TiKV Detail page on Grafana [#1071](#)

14.9.14 TiDB 3.0.7 Release Notes

Release date: December 4, 2019

TiDB version: 3.0.7

TiDB Ansible version: 3.0.7

14.9.14.1 TiDB

- Fix the issue that the lock TTL's value is too large because the TiDB server's local time is behind PD's timestamp [#13868](#)
- Fix the issue that the timezone is incorrect after parsing the date from strings using `gotime.Local` [#13793](#)
- Fix the issue that the result might be incorrect because the `binSearch` function does not return an error in the implementation of `builtinIntervalRealSig` [#13767](#)
- Fix the issue that data is incorrect because the precision is lost when an integer is converted to an unsigned floating point or decimal type [#13755](#)
- Fix the issue that the result is incorrect because the `not null` flag is not properly reset when the `USING` clause is used in Natural Outer Join and Outer Join [#13739](#)
- Fix the issue that the statistics are not accurate because a data race occurs when statistics are updated [#13687](#)

14.9.14.2 TiKV

- Make the deadlock detector only observe valid Regions to make sure the deadlock manager is in a valid Region [#6110](#)
- Fix a potential memory leak issue [#6128](#)

14.9.15 TiDB 3.0.6 Release Notes

Release date: November 28, 2019

TiDB version: 3.0.6

TiDB Ansible version: 3.0.6

14.9.15.1 TiDB

- SQL Optimizer
 - Fix the issue that the result is incorrect after the window function AST restores SQL text, for example, `over w` being mistakenly restored to `over (w)` [#12933](#)
 - Fix the issue of pushing down STREAM AGG() to doubleRead [#12690](#)
 - Fix the issue that quotes are incorrectly handled for SQL binding [#13117](#)
 - Optimize the `select max(_tidb_rowid) from t` scenario to avoid full table scans [#13095](#)
 - Fix the issue that the query result is incorrect when the query statement contains a variable assignment expression [#13231](#)
 - Fix the issue that the result is incorrect when the UPDATE statement contains both a sub-query and a generated column; fix the UPDATE statement execution error when this statement contains two same-named tables from different source databases [#13350](#)
 - Support `_tidb_rowid` for point queries [#13416](#)
 - Fix the issue that the generated query execution plan is incorrect, caused by incorrect usage of partitioned table statistics [#13628](#)
- SQL Execution Engine
 - Fix the issue that TiDB is incompatible with MySQL when handling invalid values of the year type [#12745](#)
 - Reuse Chunk in the `INSERT ON DUPLICATE UPDATE` statement to reduce the memory overhead [#12998](#)
 - Add the support for the `JSON_VALID` built-in function [#13133](#)
 - Support executing `ADMIN CHECK TABLE` on partitioned tables [#13140](#)
 - Fix the panic issue when `FAST ANALYZE` is executed on empty tables [#13343](#)
 - Fix the panic issue when executing `FAST ANALYZE` on an empty table that contains multi-column indexes [#13394](#)
 - Fix the issue that the estimated number of rows is greater than 1 when the `WHERE` clause contains an equal condition on the unique key [#13382](#)
 - Fix the issue that the returned data might be duplicated when Streaming is enabled in TiDB [#13254](#)
 - Extract the top N values from the count-min sketch to improve the estimation accuracy [#13429](#)
- Server
 - Make requests sent to TiKV fail quickly when the gRPC dial times out [#12926](#)
 - Add the following virtual tables: [#13009](#)
 - * `performance_schema.tidb_profile_allocs`
 - * `performance_schema.tidb_profile_block`
 - * `performance_schema.tidb_profile_cpu`
 - * `performance_schema.tidb_profile_goroutines`

- Fix the issue that the `kill` command does not work when the query is waiting for pessimistic locking [#12989](#)
- Do not do asynchronous rollback when acquiring pessimistic locking fails and the transaction only involves modifying a single key [#12707](#)
- Fix the panic issue when the response for the request of splitting Regions is empty [#13092](#)
- Avoid unnecessary backoff when `PessimisticLock` returns a locking error [#13116](#)
- Modify the TiDB behavior for checking configurations by printing a warning log for unrecognized configuration option [#13272](#)
- Support obtaining the binlog status of all TiDB nodes via the `/info/all` interface [#13187](#)
- Fix the issue that goroutine might leak when TiDB kills connections [#13251](#)
- Make the `innodb_lock_wait_timeout` parameter work in pessimistic transactions to control the lock wait timeout for pessimistic locking [#13165](#)
- Stop updating pessimistic transaction TTL when pessimistic transactional queries are killed to prevent other transactions from waiting unnecessarily [#13046](#)

- DDL

- Fix the issue that the execution result of `SHOW CREATE VIEW` in TiDB is inconsistent with that in MySQL [#12912](#)
- Support creating View based on `union`, for example, `create view v as select ↪ * from t1 union select * from t2` [#12955](#)
- Add more transaction-related fields for the `slow_query` table: [#13072](#)
 - * `Prewrite_time`
 - * `Commit_time`
 - * `Get_commit_ts_time`
 - * `Commit_backoff_time`
 - * `Backoff_types`
 - * `Resolve_lock_time`
 - * `Local_latch_wait_time`
 - * `Write_key`
 - * `Write_size`
 - * `Prewrite_region`
 - * `Txn_retry`
- Use the table’s `COLLATE` instead of the system’s default charset in the column when a table is created and the table contains `COLLATE` [#13174](#)
- Limit the length of the index name when creating a table [#13310](#)
- Fix the issue that the table name length is not checked when a table is renamed [#13346](#)
- Add the `alter-primary-key` configuration (disabled by default) to support adding/dropping the primary key in TiDB [#13522](#)

14.9.15.2 TiKV

- Fix the issue that the `acquire_pessimistic_lock` interface returns a wrong `txn_size` [#5740](#)
- Limit the writes for GC worker per second to reduce the impact on the performance [#5735](#)
- Make `lock_manager` accurate [#5845](#)
- Support `innodb_lock_wait_timeout` for pessimistic locking [#5848](#)
- Add the configuration check for Titan [#5720](#)
- Support using tikv-ctl to dynamically modify the GC I/O limit: `tikv-ctl --host=→ ip:port modify-tikv-config -m server -n gc.max_write_bytes_per_sec -→ v 10MB` [#5957](#)
- Reduce useless `clean` up requests to decrease the pressure on the deadlock detector [#5965](#)
- Avoid reducing TTL in pessimistic locking prewrite requests [#6056](#)
- Fix the issue that a missing blob file might occur in Titan [#5968](#)
- Fix the issue that `RocksDBOptions` might not take effect in Titan [#6009](#)

14.9.15.3 PD

- Add an `ActOn` dimension for each filter to indicate that each scheduler and checker is affected by the filter, and delete two unused filters: `disconnectFilter` and `rejectLeaderFilter` [#1911](#)
- Print a warning log when it takes more than 5 milliseconds to generate a timestamp in PD [#1867](#)
- Lower the client log level when passing unavailable endpoint to the client [#1856](#)
- Fix the issue that the gRPC message package might exceed the maximum size in the `region_syncer` replication process [#1952](#)

14.9.15.4 Tools

- TiDB Binlog
 - Obtain the initial replication timestamp from PD when `initial-commit-ts` is set to “-1” in Drainer [#788](#)
 - Decouple Drainer’s `Checkpoint` storage from the downstream and support saving `Checkpoint` in MySQL or local files [#790](#)
 - Fix the Drainer panic issue caused by using empty values when configuring replication database/table filtering [#801](#)
 - Fix the issue that processes get into the deadlock status instead of exiting after a panic occurs because Drainer fails to apply binlog files to the downstream [#807](#)
 - Fix the issue that Pump blocks when it exits because of gRPC’s `GracefulStop` [#817](#)
 - Fix the issue that Drainer fails when it receives a binlog which misses a column during the execution of a `DROP COLUMN` statement in TiDB (v3.0.6 or later) [#827](#)
- TiDB Lightning

- Add the `max-allowed-packet` configuration (64 M by default) for the TiDB backend [#248](#)

14.9.16 TiDB 3.0.5 Release Notes

Release date: October 25, 2019

TiDB version: 3.0.5

TiDB Ansible version: 3.0.5

14.9.16.1 TiDB

- SQL Optimizer
 - Support boundary checking on Window Functions [#12404](#)
 - Fix the issue that `IndexJoin` on the partition table returns incorrect results [#12712](#)
 - Fix the issue that the `ifnull` function on the top of the outer join `Apply` operator returns incorrect results [#12694](#)
 - Fix the issue of update failure when a subquery was included in the `where` condition of `UPDATE` [#12597](#)
 - Fix the issue that outer join was incorrectly converted to inner join when the `cast` function was included in the query conditions [#12790](#)
 - Fix incorrect expression passing in the join condition of `AntiSemiJoin` [#12799](#)
 - Fix the statistics error caused by shallow copy when initializing statistics [#12817](#)
 - Fix the issue that the `str_to_date` function in TiDB returns a different result from MySQL when the date string and the format string do not match [#12725](#)
- SQL Execution Engine
 - Fix the panic issue when the `from_unixtime` function handles null [#12551](#)
 - Fix the `invalid list index` error reported when canceling DDL jobs [#12671](#)
 - Fix the issue that arrays were out of bounds when Window Functions are used [#12660](#)
 - Improve the behavior of the `AutoIncrement` column when it is implicitly allocated, to keep it consistent with the default mode of MySQL auto-increment locking (“consecutive” lock mode): for the implicit allocation of multiple `AutoIncrement` IDs in a single-line `Insert` statement, TiDB guarantees the continuity of the allocated values. This improvement ensures that the JDBC `getGeneratedKeys()` method will get the correct results in any scenario. [#12602](#)
 - Fix the issue that the query is hanged when `HashAgg` serves as a child node of `Apply` [#12766](#)
 - Fix the issue that the `AND` and `OR` logical expressions return incorrect results when it comes to type conversion [#12811](#)
- Server

- Implement the interface function that modifies transaction TTL to help support large transactions later [#12397](#)
- Support extending the transaction TTL as needed (up to 10 minutes) to support pessimistic transactions [#12579](#)
- Adjust the number of times that TiDB caches schema changes and corresponding changed table information from 100 to 1024, and support modification by using the `tidb_max_delta_schema_count` system variable [#12502](#)
- Update the behavior of the `kvrpc.Cleanup` protocol to no longer clean locks of transactions that are not overtime [#12417](#)
- Support logging Partition table information to the `information_schema.tables` table [#12631](#)
- Support modifying the TTL of Region Cache by configuring `region-cache-ttl` [#12683](#)
- Support printing the execution plan compression-encoded information in the slow log. This feature is enabled by default and can be controlled by using the `slow->log-plan` configuration or the `tidb_record_plan_in_slow_log` variable. In addition, the `tidb_decode_plan` function can decode the execution plan column encoded information in the slow log into execution plan information. [#12808](#)
- Support displaying memory usage information in the `information_schema.
processlist` table [#12801](#)
- Fix the issue that an error and an unexpected alarm might occur when the TiKV Client judges an idle connection [#12846](#)
- Fix the issue that the `INSERT IGNORE` statement performance is decreased because `tikvSnapshot` does not properly cache the KV results of `BatchGet()` [#12872](#)
- Fix the issue that the TiDB response speed was relatively low because of slow connection to some KV services [#12814](#)

- DDL

- Fix the issue that the `Create Table` operation does not correctly set the Int type default value for the Set column [#12267](#)
- Support multiple `uniques` when creating a unique index in the `Create Table` statement [#12463](#)
- Fix the issue that populating the default value of this column for existing rows might cause an error when adding a Bit type column using `Alter Table` [#12489](#)
- Fix the failure of adding a partition when the Range partitioned table uses a Date or Datetime type column as the partitioning key [#12815](#)
- Support checking the consistency of the partition type and the partition key type when creating a table or adding a partition, for the Range partitioned table with the Date or Datetime type column as the partition key [#12792](#)
- Add a check that the Unique Key column set needs to be greater than or equal to the partitioned column set when creating a Range partitioned table [#12718](#)

- Monitor

- Add the monitoring metrics of Commit and Rollback operations to the `Transaction OPS` dashboard [#12505](#)

- Add the monitoring metrics of Add Index operation progress [#12390](#)

14.9.16.2 TiKV

- Storage
 - Add a new feature of pessimistic transactions: the transaction cleanup interface supports only cleaning up locks whose TTL is outdated [#5589](#)
 - Fix the issue that Rollback of the transaction Primary key is collapsed [#5646](#), [#5671](#)
 - Fix the issue that under pessimistic locks, point queries might return the previous version data [#5634](#)
- Raftstore
 - Reduce message flush operations in Raftstore to improve performance and reduce CPU usage [#5617](#)
 - Optimize the cost of obtaining the Region size and estimated number of keys, to reduce heartbeat overhead and CPU usage [#5620](#)
 - Fix the issue that Raftstore prints an error log and encounters a panic when getting invalid data [#5643](#)
- Engine
 - Enable RocksDB `force_consistency_checks` to improve data safety [#5662](#)
 - Fix the issue that concurrent flush operations in Titan might cause data loss [#5672](#)
 - Update the rust-rocksdb version to avoid the issue of TiKV crash and restart caused by intra-L0 compaction [#5710](#)

14.9.16.3 PD

- Improve the precision of storage occupied by Regions [#1782](#)
- Improve the output of the `--help` command [#1763](#)
- Fix the issue that the HTTP request fails to redirect after TLS is enabled [#1777](#)
- Fix the panic issue occurred when pd-ctl uses the `store shows limit` command [#1808](#)
- Improve readability of label monitoring metrics and reset the original leader's monitoring data when the leader switches, to avoid false reports [#1815](#)

14.9.16.4 Tools

- TiDB Binlog
 - Fix the issue that `ALTER DATABASE` related DDL operations cause Drainer to exit abnormally [#769](#)

- Support querying the transaction status information for Commit binlog to improve replication efficiency [#757](#)
- Fix the issue that a Pump panic might occur when Drainer's `start_ts` is greater than Pump's largest `commit_ts` [#758](#)
- TiDB Lightning
 - Integrate the full logic import feature of Loader and support configuring the back-end mode [#221](#)

14.9.16.5 TiDB Ansible

- Add the monitoring metrics of adding index speed [#986](#)
- Simplify the configuration file content and remove parameters that users do not need to configure [#1043c](#), [#998](#)
- Fix the monitoring expression error of performance read and performance write [#e90e7](#)
- Update the monitoring display method and the alarm rules of Raftstore CPU usage [#992](#)
- Update the TiKV CPU monitoring item in the Overview monitoring dashboard to filter out the excess monitoring content [#1001](#)

14.9.17 TiDB 3.0.4 Release Notes

Release date: October 8, 2019

TiDB version: 3.0.4

TiDB Ansible version: 3.0.4

- New features
 - Add the `performance_schema.events_statements_summary_by_digest` system table to troubleshoot performance issues at the SQL level
 - Add the `WHERE` clause in TiDB's `SHOW TABLE REGIONS` syntax
 - Add the `worker-count` and `txn-batch` configuration items in Reparo to control the recovery speed
- Improvements
 - Support batch Region split command and empty split command in TiKV to improve split performance
 - Support double linked list for RocksDB in TiKV to improve performance of reverse scan
 - Add two perf tools `iosnoop` and `funcslower` in TiDB Ansible to better diagnose the cluster state
 - Optimize the output of slow query logs in TiDB by deleting redundant fields

- Changed behaviors
 - Update the default value of `txn-local-latches.enable` to `false` to disable the default behavior of checking conflicts of local transactions in TiDB
 - Add the `tidb_txn_mode` system variable of global scope in TiDB and allow using the pessimistic lock; note that TiDB still adopts the optimistic lock by default
 - Replace the `Index_ids` field in TiDB slow query logs with `Index_names` to improve the usability of slow query logs
 - Add the `split-region-max-num` parameter in the TiDB configuration file to modify the maximum number of Regions allowed in the `SPLIT TABLE` syntax
 - Return the `Out Of Memory Quota` error instead of disconnecting the link when a SQL execution exceeds the memory limit
 - Disallow dropping the `AUTO_INCREMENT` attribute of columns in TiDB to avoid misoperations. To drop this attribute, change the `tidb_allow_remove_auto_inc` system variable
- Fixed issues
 - Fix the issue that the uncommented TiDB-specific syntax `PRE_SPLIT_REGIONS` might cause errors in the downstream database during data replication
 - Fix the issue in TiDB that the slow query logs are incorrect when getting the result of `PREPARE + EXECUTE` by using the cursor
 - Fix the issue in PD that adjacent small Regions cannot be merged
 - Fix the issue in TiKV that file descriptor leak in idle clusters might cause TiKV processes to exit abnormally when the processes run for a long time
- Contributors

Our thanks go to the following contributors from the community for helping this release:

- [sduzh](#)
- [lizhenda](#)

14.9.17.1 TiDB

- SQL Optimizer
 - Fix the issue that invalid query ranges might be resulted when splitted by feedback [#12170](#)
 - Display the returned error of the `SHOW STATS_BUCKETS` statement in hexadecimal rather than return errors when the result contains invalid Keys [#12094](#)
 - Fix the issue that when a query contains the `SLEEP` function (for example, `select → 1 from (select sleep(1))t;`), column pruning causes invalid `sleep(1)` during query [#11953](#)
 - Use index scan to lower IO when a query only concerns the number of columns rather than the table data [#12112](#)

- Do not use any index when no index is specified in `use index()` to be compatible with MySQL [#12100](#)
- Strictly limit the number of TopN records in the `CMSketch` statistics to fix the issue that the `ANALYZE` statement fails because the statement count exceeds TiDB's limit on the size of a transaction [#11914](#)
- Fix the error occurred when converting the subqueries contained in the `Update` statement [#12483](#)
- Optimize execution performance of the `select ... limit ... offset ...` statement by pushing the Limit operator down to the `IndexLookUpReader` execution logic [#12378](#)
- SQL Execution Engine
 - Print the SQL statement in the log when the `PREPARED` statement is incorrectly executed [#12191](#)
 - Support partition pruning when the `UNIX_TIMESTAMP` function is used to implement partitioning [#12169](#)
 - Fix the issue that no error is reported when `AUTO_INCREMENT` incorrectly allocates `MAX int64` and `MAX uint64` [#12162](#)
 - Add the `WHERE` clause in the `SHOW TABLE ... REGIONS` and `SHOW TABLE .. INDEX ↵ ... REGIONS` syntaxes [#12123](#)
 - Return the `Out Of Memory Quota` error instead of disconnecting the link when a SQL execution exceeds the memory limit [#12127](#)
 - Fix the issue that incorrect result is returned when `JSON_UNQUOTE` function handles JSON text [#11955](#)
 - Fix the issue that `LAST INSERT ID` is incorrect when assigning values to the `AUTO_INCREMENT` column in the first row (for example, `insert into t (pk, ↵ c)values (1, 2), (NULL, 3)`) [#12002](#)
 - Fix the issue that the `GROUPBY` parsing rule is incorrect in the `PREPARE` statement [#12351](#)
 - Fix the issue that the privilege check is incorrect in the point queries [#12340](#)
 - Fix the issue that the duration by `sql_type` for the `PREPARE` statement is not shown in the monitoring record [#12331](#)
 - Support using aliases for tables in the point queries (for example, `select * from ↵ t tmp where a = "aa"`) [#12282](#)
 - Fix the error occurred when not handling negative values as unsigned when inserting negative numbers into BIT type columns [#12423](#)
 - Fix the incorrectly rounding of time (for example, `2019-09-11 11:17:47.999999666 ↵` should be rounded to `2019-09-11 11:17:48.`) [#12258](#)
 - Refine the usage of expression blocklist (for example, `<` is equivalent to `It.`) [#11975](#)
 - Add the database prefix to the message of non-existing function error (for example, `[expression:1305]FUNCTION test.std_samp does not exist`) [#12111](#)
- Server
 - Add the `Prev_stmt` field in slow query logs to output the previous statement

when the last statement is COMMIT [#12180](#)

- Optimize the output of slow query logs by deleting redundant fields [#12144](#)
- Update the default value of `txn-local-latches.enable` to `false` to disable the default behavior of checking conflicts of local transactions in TiDB [#12095](#)
- Replace the `Index_ids` field in TiDB slow query logs with `Index_names` to improve the usability of slow query logs [#12061](#)
- Add the `tidb_txn_mode` system variable of global scope in TiDB and allow using pessimistic lock [#12049](#)
- Add the `Backoff` field in the slow query logs to record the Backoff information in the commit phase of 2PC [#12335](#)
- Fix the issue that the slow query logs are incorrect when getting the result of PREPARE + EXECUTE by using the cursor (for example, PREPARE stmt1FROM
→ SELECT * FROM t WHERE a > ?; EXECUTE stmt1 USING @variable) [#12392](#)
- Support `tidb_enable_stmt_summary`. When this feature is enabled, TiDB counts the SQL statements and the result can be queried by using the system table `performance_schema.events_statements_summary_by_digest` [#12308](#)
- Adjust the level of some logs in tikv-client (for example, change the log level of `batchRecvLoop` fails from ERROR to INFO) [#12383](#)

- DDL

- Add the `tidb_allow_remove_auto_inc` variable. Dropping the AUTO INCREMENT attribute of the column is disabled by default [#12145](#)
- Fix the issue that the uncommented TiDB-specific syntax `PRE_SPLIT_REGIONS` might cause errors in the downstream database during data replication [#12120](#)
- Add the `split-region-max-num` variable in the configuration file so that the maximum allowable number of Regions is adjustable [#12097](#)
- Support splitting a Region into multiple Regions and fix the timeout issue during Region scatterings [#12343](#)
- Fix the issue that the `drop index` statement fails when the index that contains an AUTO_INCREMENT column referenced by two indexes [#12344](#)

- Monitor

- Add the `connection_transient_failure_count` monitoring metrics to count the number of gRPC connection errors in `tikvclient` [#12093](#)

14.9.17.2 TiKV

- Raftstore

- Fix the issue that Raftstore inaccurately counts the number of keys in empty Regions [#5414](#)
- Support double linked list for RocksDB to improve the performance of reverse scan [#5368](#)

- Support batch Region split command and empty split command to improve split performance [#5470](#)
- Server
 - Fix the issue that the output format of the -V command is not consistent with the format of 2.X [#5501](#)
 - Upgrade Titan to the latest version in the 3.0 branch [#5517](#)
 - Upgrade grpcio to v0.4.5 [#5523](#)
 - Fix the issue of gRPC coredump and support shared memory to avoid OOM [#5524](#)
 - Fix the issue in TiKV that file descriptor leak in idle clusters might cause TiKV processes to exit abnormally when the processes run for a long time [#5567](#)
- Storage
 - Support the `txn_heart_beat` API to make the pessimistic lock in TiDB consistent with that in MySQL as much as possible [#5507](#)
 - Fix the issue that the performance of point queries is low in some situations [#5495](#) [#5463](#)

14.9.17.3 PD

- Fix the issue that adjacent small Regions cannot be merged [#1726](#)
- Fix the issue that the TLS enabling parameter in pd-ctl is invalid [#1738](#)
- Fix the thread-safety issue that the PD operator is accidentally removed [#1734](#)
- Support TLS for Region syncer [#1739](#)

14.9.17.4 Tools

- TiDB Binlog
 - Add the `worker-count` and `txn-batch` configuration items in Reparo to control the recovery speed [#746](#)
 - Optimize the memory usage of Drainer to enhance the efficiency of simultaneous execution [#737](#)
- TiDB Lightning
 - Fix the issue that re-importing data from checkpoint might cause TiDB Lightning to panic [#237](#)
 - Optimize the algorithm of `AUTO_INCREMENT` to reduce the risk of overflowing `AUTO_INCREMENT` columns [#227](#)

14.9.17.5 TiDB Ansible

- Upgrade TiSpark to v2.2.0 [#926](#)
- Update the default value of the TiDB configuration item `pessimistic_txn` to `true` [#933](#)
- Add more system-level monitoring metrics to `node_exporter` [#938](#)
- Add two perf tools `iosnoop` and `funcslower` in TiDB Ansible to better diagnose the cluster state [#946](#)
- Replace the raw module to shell module to address the long waiting time in such situations as the password expires [#949](#)
- Update the default value of the TiDB configuration item `txn_local_latches` to `false`
- Optimize the monitoring metrics and alert rules of Grafana dashboard [#962](#) [#963](#) [#969](#)
- Check the configuration file before the deployment and upgrade [#934](#) [#972](#)

14.9.18 TiDB 3.0.3 Release Notes

Release date: August 29, 2019

TiDB version: 3.0.3

TiDB Ansible version: 3.0.3

14.9.18.1 TiDB

- SQL Optimizer
 - Add the `opt_rule_blacklist` table to disable logic optimization rules such as `aggregation_eliminate` and `column_prune` [#11658](#)
 - Fix the issue that incorrect results might be returned for `Index Join` when the join key uses a prefix index or an unsigned index column that is equal to a negative value [#11759](#)
 - Fix the issue that ” or \ in the SELECT statements of `create ... binding ...` might result in parsing errors [#11726](#)
- SQL Execution Engine
 - Fix the issue that type errors in the return value might occur when the `Quote` function handles a null value [#11619](#)
 - Fix the issue that incorrect results for `ifnull` might be returned when Max/Min is used for type inferring with `NotNullFlag` retained [#11641](#)
 - Fix the potential error that occurs when comparing bit type data in string form [#11660](#)
 - Decrease the concurrency for data that requires sequential read to lower the possibility of OOM [#11679](#)

- Fix the issue that incorrect type inferring might be caused when multiple parameters are unsigned for some built-in functions (for example, `if` and `coalesce`) [#11621](#)
- Fix the incompatibility with MySQL when the `Div` function handles unsigned decimal types [#11813](#)
- Fix the issue that panic might occur when executing SQL statements that modify the status of Pump/Drainer [#11827](#)
- Fix the issue that panic might occur for `select ... for update` when Autocommit = 1 and there is no `begin` statement [#11736](#)
- Fix the permission check error that might occur when the `set default role` statement is executed [#11777](#)
- Fix the permission check error that might occur when `create user` or `drop user` is executed [#11814](#)
- Fix the issue that the `select ... for update` statement might auto retry when it is constructed into the `PointGetExecutor` function [#11718](#)
- Fix the boundary error that might occur when the Window function handles partition [#11825](#)
- Fix the issue that the `time` function hits EOF errors when handling an incorrectly formatted argument [#11893](#)
- Fix the issue that the Window function does not check the passed-in parameters [#11705](#)
- Fix the issue that the plan result viewed via `Explain` is inconsistent with the actually executed plan [#11186](#)
- Fix the issue that duplicate memory referenced by the Window function might result in a crash or incorrect results [#11823](#)
- Update the incorrect information in the `Succ` field in the slow log [#11887](#)
- Server
 - Rename the `tidb_back_off_wexight` variable to `tidb_backoff_weight` [#11665](#)
 - Update the minimum TiKV version compatible with the current TiDB to v3.0.0 [#11618](#)
 - Support `make testSuite` to ensure the suites in the test are correctly used [#11685](#)
- DDL
 - Skip the execution of unsupported partition-related DDL statements, including statements that modify the partition type while deleting multiple partitions [#11373](#)
 - Disallow a Generated Column to be placed before its dependent columns [#11686](#)
 - Modify the default values of `tidb_ddl_reorg_worker_cnt` and `tidb_ddl_reorg_batch_size` ↳ [#11874](#)
- Monitor
 - Add new backoff monitoring types to record duration for each backoff type; add more backoff metrics to cover previously uncounted types such as commit backoff

#11728

14.9.18.2 TiKV

- Fix the issue that ReadIndex might fail to respond to requests because of duplicate context [#5256](#)
- Fix potential scheduling jitters caused by premature PutStore [#5277](#)
- Fix incorrect timestamps reported from Region heartbeats [#5296](#)
- Reduce the size of core dump by excluding the shared block cache from it [#5322](#)
- Fix potential TiKV panics during region merge [#5291](#)
- Speed up leader change check for the dead lock detector [#5317](#)
- Support using `grpc env` to create deadlock clients [#5346](#)
- Add `config-check` to check whether the configuration is correct [#5349](#)
- Fix the issue that ReadIndex does not return anything when there is no leader [#5351](#)

14.9.18.3 PD

- Return success message for `pdctl` [#1685](#)

14.9.18.4 Tools

- TiDB Binlog
 - Modify the default value of `defaultBinlogItemCount` in Drainer from 65536 to 512 to reduce the chance of OOM on Drainer startup [#721](#)
 - Optimize the offline logic for pump server to avoid potential offline congestions [#701](#)
- TiDB Lightning:
 - Skip the system databases `mysql`, `information_schema`, `performance_schema`, and `sys` by default when importing [#225](#)

14.9.18.5 TiDB Ansible

- Optimize PD operations for rolling update to improve stability [#894](#)
- Remove the Grafana Collector components that are not supported by the current Grafana version [#892](#)
- Update TiKV alerting rules [#898](#)
- Fix the issue that the generated TiKV configuration misses the `pessimistic-txn` parameter [#911](#)
- Update Spark to V2.4.3, and update TiSpark to V2.1.4 that is compatible with Spark V2.4.3 [#913](#), [#918](#)

14.9.19 TiDB 3.0.2 Release Notes

Release date: August 7, 2019

TiDB version: 3.0.2

TiDB Ansible version: 3.0.2

14.9.19.1 TiDB

- SQL Optimizer

- Fix the issue that the “Can’t find column in schema” message is reported when the same table occurs multiple times in a query and logically the query result is always empty [#11247](#)
- Fix the issue that the query plan does not meet the expectation caused by the `TIDB_INLJ` hint not working correctly in some cases (like `explain select /*+ ↳ TIDB_INLJ(t1)*/ t1.b, t2.a from t t1, t t2 where t1.b = t2.a`) [#11362](#)
- Fix the issue that the column name in the query result is wrong in some cases (like `SELECT IF(1,c,c)FROM t`) [#11379](#)
- Fix the issue that some queries like `SELECT 0 LIKE 'a string'` return TRUE because the `LIKE` expression is implicitly converted to 0 in some cases [#11411](#)
- Support sub-queries in the `SHOW` statement, like `SHOW COLUMNS FROM tbl WHERE ↳ FIELDS IN (SELECT 'a')` [#11459](#)
- Fix the issue that the related column of the aggregate function cannot be found and an error is reported caused by the `outerJoinElimination` optimizing rule not correctly handling the column alias; improve alias parsing in the optimizing process to make optimization cover more query types [#11377](#)
- Fix the issue that no error is reported when the syntax restriction is violated in the Window function (for example, `UNBOUNDED PRECEDING` is not allowed to appear at the end of the Frame definition) [#11543](#)
- Fix the issue that `FUNCTION_NAME` is in uppercase in the `ERROR 3593 (HY000)`:
 ↳ You cannot use the window function `FUNCTION_NAME` in this context error message, which causes incompatibility with MySQL [#11535](#)
- Fix the issue that the unimplemented `IGNORE NULLS` syntax in the Window function is used but no error is reported [#11593](#)
- Fix the issue that the Optimizer does not correctly estimate time equal conditions [#11512](#)
- Support updating the Top-N statistics based on the feedback information [#11507](#)

- SQL Execution Engine

- Fix the issue that the returned value is not `NULL` when the `INSERT` function contains `NULL` in parameters [#11248](#)
- Fix the issue that the computing result might be wrong when the partitioned table is checked by the `ADMIN CHECKSUM` operation [#11266](#)

- Fix the issue that the result might be wrong when INDEX JOIN uses the prefix index [#11246](#)
- Fix the issue that result might be wrong caused by incorrectly aligning fractions when the DATE_ADD function does subtraction on date numbers involving microseconds [#11288](#)
- Fix the wrong result caused by the DATE_ADD function incorrectly processing the negative numbers in INTERVAL [#11325](#)
- Fix the issue that the number of fractional digits returned by Mod(%), Multiple ↪ (*) or Minus(-) is different from that in MySQL when Mod(%), Multiple(*) or Minus(-) returns 0 and the number of fractional digits is large (like select ↪ 0.000 % 0.11234500000000000000) [#11251](#)
- Fix the issue that NULL with a warning is incorrectly returned when the length of the result returned by CONCAT and CONCAT_WS functions exceeds max_allowed_packet [#11275](#)
- Fix the issue that NULL with a warning is incorrectly returned when parameters in the SUBTIME and ADDTIME functions are invalid [#11337](#)
- Fix the issue that NULL is incorrectly returned when parameters in the CONVERT_TZ function are invalid [#11359](#)
- Add the MEMORY column to the result returned by EXPLAIN ANALYZE to show the memory usage of this query [#11418](#)
- Add CARTESIAN Join to the result of EXPLAIN [#11429](#)
- Fix the incorrect data of auto-increment columns of the float and double types [#11385](#)
- Fix the panic issue caused by some nil information when pseudo statistics are dumped [#11460](#)
- Fix the incorrect query result of SELECT ... CASE WHEN ... ELSE NULL ... caused by constant folding optimization [#11441](#)
- Fix the issue that floatStrToIntStr does not correctly parse the input such as +999.9999e2 [#11473](#)
- Fix the issue that NULL is not returned in some cases when the result of the DATE_ADD and DATE_SUB function overflows [#11476](#)
- Fix the issue that the conversion result is different from that in MySQL if the string contains an invalid character when a long string is converted to an integer [#11469](#)
- Fix the issue that the result of the REGEXP BINARY function is incompatible with MySQL caused by case sensitiveness of this function [#11504](#)
- Fix the issue that an error is reported when the GRANT ROLE statement receives CURRENT_ROLE; fix the issue that the REVOKE ROLE statement does not correctly revoke the mysql.default_role privilege [#11356](#)
- Fix the display format issue of the Incorrect datetime value warning information when executing statements like SELECT ADDDATE('2008-01-34', -1) [#11447](#)
- Fix the issue that the error message reports constant ... overflows float ↪ rather than constant ... overflows bigint if the result overflows when a float field of the JSON data is converted to an integer [#11534](#)

- Fix the issue that the result might be wrong caused by incorrect type conversion when the DATE_ADD function receives FLOAT, DOUBLE and DECIMAL column parameters [#11527](#)
- Fix the wrong result caused by incorrectly processing the sign of the INTERVAL fraction in the DATE_ADD function [#11615](#)
- Fix the incorrect query result when Index Lookup Join contains the prefix index caused by Ranger not correctly handling the prefix index [#11565](#)
- Fix the issue that the “Incorrect arguments to NAME_CONST” message is reported if the NAME_CONST function is executed when the second parameter of NAME_CONST is a negative number [#11268](#)
- Fix the issue that the result is incompatible with MySQL when an SQL statement involves computing the current time and the value is fetched multiple times; use the same value when fetching the current time for the same SQL statement [#11394](#)
- Fix the issue that Close is not called for ChildExecutor when the Close of baseExecutor reports an error. This issue might lead to Goroutine leaks when the KILL statements do not take effect and ChildExecutor is not closed [#11576](#)
- Server
 - Fix the issue that the auto-added value is 0 instead of the current timestamp when LOAD DATA processes the missing TIMESTAMP field in the CSV file [#11250](#)
 - Fix issues that the SHOW CREATE USER statement does not correctly check related privileges, and USER and HOST returned by SHOW CREATE USER CURRENT_USER() might be wrong [#11229](#)
 - Fix the issue that the returned result might be wrong when executeBatch is used in JDBC [#11290](#)
 - Reduce printing the log information of the streaming client when changing the TiKV server’s port [#11370](#)
 - Optimize the logic of reconnecting the streaming client to the TiKV server so that the streaming client will not be blocked for a long time [#11372](#)
 - Add REGION_ID in INFORMATION_SCHEMA.TIDB_HOT_REGIONS [#11350](#)
 - Cancel the timeout duration of obtaining Region information from the PD API to ensure that obtaining Region information will not end in a failure when TiDB API `http://{TiDBIP}:10080/regions/hot` is called due to PD timeout when the number of Regions is large [#11383](#)
 - Fix the issue that Region related requests do not return partitioned table-related Regions in the HTTP API [#11466](#)
 - Make the following changes to reduce the probability of locking timeout caused by slow operations when the user manually validates pessimistic locking [#11521](#):
 - * Increase the default TTL of pessimistic locking from 30 seconds to 40 seconds
 - * Increase the maximum TTL from 60 seconds to 120 seconds
 - * Calculate the pessimistic locking duration from the first LockKeys request
 - Change the SendRequest function logic in the TiKV client: try to immediately connect to another peer instead of keeping waiting when the connect cannot be built [#11531](#)

- Optimize the Region cache: label the removed store as invalid when a store is moved while another store goes online with a same address, to update the store information in the cache as soon as possible [#11567](#)
- Add the Region ID to the result returned by the `http://{TiDB_ADDRESS}:{TiDB_IP} ↵ }/mvcc/key/{db}/{table}/{handle}` API [#11557](#)
- Fix the issue that Scatter Table does not work caused by the Scatter Table API not escaping the Range key [#11298](#)
- Optimize the Region cache: label the store where the Region exists as invalid when the correspondent store is inaccessible, to avoid reduced query performance caused by accessing this store [#11498](#)
- Fix the error that the table schema can still be obtained through the HTTP API after dropping the database with the same name multiple times [#11585](#)

- DDL

- Fix the issue that an error occurs when a non-string column with a zero length is being indexed [#11214](#)
- Disallow modifying the columns with foreign key constraints and full-text indexes (Note: TiDB still supports foreign key constraints and full-text indexes in syntax) [#11274](#)
- Fix the issue that the index offset of the column might be wrong because the position changed by the `ALTER TABLE` statement and the default value of the column are used concurrently [#11346](#)
- Fix two issues that occur when parsing JSON files:
 - * `int64` is used as the intermediate parsing result of `uint64` in `ConvertJSONToFloat ↵ ,` which leads to the precision overflow error [#11433](#)
 - * `int64` is used as the intermediate parsing result of `uint64` in `ConvertJSONToInt ↵ ,` which leads to the precision overflow error [#11551](#)
- Disallow dropping indexes on the auto-increment column to avoid that the auto-increment column might get an incorrect result [#11399](#)
- Fix the following issues [#11492](#):
 - * The character set and the collation of the column are not consistent when explicitly specifying the collation but not the character set
 - * The error is not correctly reported when there is a conflict between the character set and the collation that are specified by `ALTER TABLE ... MODIFY ↵ COLUMN`
 - * Incompatibility with MySQL when using `ALTER TABLE ... MODIFY COLUMN` to specify character sets and collations multiple times
- Add the trace details of the subquery to the result of the `TRACE` query [#11458](#)
- Optimize the performance of executing `ADMIN CHECK TABLE` and greatly reduce its execution time [#11547](#)
- Add the result returned by `SPLIT TABLE ... REGIONS/INDEX` and make `TOTAL_SPLIT_REGION` and `SCATTER_FINISH_RATIO` display the number of Regions that have been split successfully before timeout in the result [#11484](#)

- Fix the issue that the precision displayed by statements like `SHOW CREATE TABLE` is incomplete when `ON UPDATE CURRENT_TIMESTAMP` is the column attribute and the float precision is specified [#11591](#)
- Fix the issue that the index result of the column cannot be correctly calculated when the expression of a virtual generated column contains another virtual generated column [#11475](#)
- Fix the issue that the minus sign cannot be added after `VALUE LESS THAN` in the `ALTER TABLE ... ADD PARTITION ...` statement [#11581](#)
- Monitor
 - Fix the issue that data is not collected and reported because the `TiKVTxnCmdCounter` → monitoring metric is not registered [#11316](#)
 - Add the `BindUsageCounter`, `BindTotalGauge` and `BindMemoryUsage` monitoring metrics for the Bind Info [#11467](#)

14.9.19.2 TiKV

- Fix the bug that TiKV panics if the Raft log is not written in time [#5160](#)
- Fix the bug that the panic information is not written into the log file after TiKV panics [#5198](#)
- Fix the bug that the Insert operation might be incorrectly performed in the pessimistic transaction [#5203](#)
- Lower the output level of some logs that require no manual intervention to INFO [#5193](#)
- Improve the accuracy of monitoring the storage engine size [#5200](#)
- Improve the accuracy of the Region size in tikv-ctl [#5195](#)
- Improve the performance of the deadlock detector for pessimistic locking [#5192](#)
- Improve the performance of GC in the Titan storage engine [#5197](#)

14.9.19.3 PD

- Fix the bug that the Scatter Region scheduler cannot work [#1642](#)
- Fix the bug that the merge Region operation cannot be performed in pd-ctl [#1653](#)
- Fix the bug that the remove-tombstone operation cannot be performed in pd-ctl [#1651](#)
- Fix the issue that the Region overlapping with the key scope cannot be found when performing the scan Region operation [#1648](#)
- Add the retrying mechanism to make sure that the members are added successfully in PD [#1643](#)

14.9.19.4 Tools

TiDB Binlog

- Add the configuration item check feature when starting, which will stop the Binlog service and report an error when an invalid item is found [#687](#)

- Add the `node-id` configuration in Drainer to specify a specific logic used by Drainer [#684](#)

TiDB Lightning

- Fix the issue that `tikv_gc_life_time` fails to be changed back to its original value when 2 checksums are running at the same time [#218](#)
- Add the configuration item check feature when starting, which will stop the Binlog service and report an error when an invalid item is found [#217](#)

14.9.19.5 TiDB Ansible

- Fix the unit error that the Disk Performance monitor treats seconds as milliseconds [#840](#)
- Add the `log4j` configuration file in Spark [#841](#)
- Fix the issue that the Prometheus configuration file is generated in the wrong format when Binlog is enabled and Kafka or ZooKeeper is configured [#844](#)
- Fix the issue that the `pessimistic-txn` configuration parameter is left out in the generated TiDB configuration file [#850](#)
- Add and optimize metrics on the TiDB Dashboard [#853](#)
- Add descriptions for each monitoring item on the TiDB Dashboard [#854](#)
- Add the TiDB Summary Dashboard to better view the cluster status and troubleshoot issues [#855](#)
- Update the Allocator Stats monitoring item on the TiKV Dashboard [#857](#)
- Fix the unit error in the Node Exporter's alerting expression [#860](#)
- Upgrade the TiSpark jar package to v2.1.2 [#862](#)
- Update the descriptions of the Ansible Task feature [#867](#)
- Update the expression of the local reader requests monitoring item on the TiDB Dashboard [#874](#)
- Update the expression of the TiKV Memory monitoring item on the Overview Dashboard, and fix the issue of wrongly displayed monitoring [#879](#)
- Remove the Binlog support in the Kafka mode [#878](#)
- Fix the issue that PD fails to transfer the Leader when executing the `rolling_update ↩ .yml` operation [#887](#)

14.9.20 TiDB 3.0.1 Release Notes

Release date: July 16, 2019

TiDB version: 3.0.1

TiDB Ansible version: 3.0.1

14.9.20.1 TiDB

- Add support for the MAX_EXECUTION_TIME feature [#11026](#)
- Add the tidb_wait_split_region_finish_backoff session variable to control the backoff time of splitting Regions [#11166](#)
- Support automatically adjusting the incremental gap allocated by auto-increment IDs based on the load, and the auto-adjustment scope of the incremental gap is 1000~2000000 [#11006](#)
- Add the ADMIN PLUGINS ENABLE/ADMIN PLUGINS DISABLE SQL statement to dynamically enable or disable plugins [#11157](#)
- Add the session connection information in the Audit plugin [#11013](#)
- Change the default behavior during the period of splitting Regions to wait for PD to finish scheduling [#11166](#)
- Prohibit Window Functions from being cached in Prepare Plan Cache to avoid incorrect results in some cases [#11048](#)
- Prohibit ALTER statements from modifying the definition of stored generated columns [#11068](#)
- Disallow changing virtual generated columns to stored generated columns [#11068](#)
- Disallow changing the generated column expression with indexes [#11068](#)
- Support compiling TiDB on the ARM64 architecture [#11150](#)
- Support modifying the collation of a database or a table, but the character set of the database/table has to be UTF-8 or utf8mb4 [#11086](#)
- Fix the issue that an error is reported when the SELECT subquery in the UPDATE ... ↪ SELECT statement fails to parse the column in the UPDATE expression and the column is wrongly pruned [#11252](#)
- Fix the panic issue that happens when a column is queried on multiple times and the returned result is NULL during point queries [#11226](#)
- Fix the data race issue caused by non-thread safe rand.Rand when using the RAND function [#11169](#)
- Fix the bug that the memory usage of a SQL statement exceeds the threshold but the execution of this statement is not canceled in some cases when oom-action="cancel" is configured, and the returned result is incorrect [#11004](#)
- Fix the issue that SHOW PROCESSLIST shows that the memory usage is not 0 because the memory usage of MemTracker was not correctly cleaned [#10970](#)
- Fix the bug that the result of comparing integers and non-integers is not correct in some cases [#11194](#)
- Fix the bug that the query result is not correct when the query on table partitions contains a predicate in explicit transactions [#11196](#)
- Fix the DDL job panic issue because infoHandle might be NULL [#11022](#)
- Fix the issue that the query result is not correct because the queried column is not referenced in the subquery and is then wrongly pruned when running a nested aggregation query [#11020](#)
- Fix the issue that the Sleep function does not respond to the KILL statement in time [#11028](#)

- Fix the issue that the DB and INFO columns shown by the SHOW PROCESSLIST command are incompatible with MySQL [#11003](#)
- Fix the system panic issue caused by the FLUSH PRIVILEGES statement when skip-
→ grant-table=true is configured [#11027](#)
- Fix the issue that the primary key statistics collected by FAST ANALYZE are not correct when the table primary key is an UNSIGNED integer [#11099](#)
- Fix the issue that the “invalid key” error is reported by the FAST ANALYZE statement in some cases [#11098](#)
- Fix the issue that the precision shown by the SHOW CREATE TABLE statement is incomplete when CURRENT_TIMESTAMP is used as the default value of the column and the float precision is specified [#11088](#)
- Fix the issue that the function name is not in lowercase when window functions report an error to make it compatible with MySQL [#11118](#)
- Fix the issue that TiDB fails to connect to TiKV and thus cannot provide service after the background thread of TiKV Client Batch gRPC panics [#11101](#)
- Fix the issue that the variable is set incorrectly by SetVar because of the shallow copy of the string [#11044](#)
- Fix the issue that the execution fails and an error is reported when the INSERT ... ON
→ DUPLICATE statement is applied on table partitions [#11231](#)
- Pessimistic locking (experimental feature)
 - Fix the issue that an incorrect result is returned because of the invalid lock on the row when point queries are run using the pessimistic locking and the returned data is empty [#10976](#)
 - Fix the issue that the query result is not correct because SELECT ... FOR UPDATE does not use the correct TSO when using the pessimistic locking in the query [#11015](#)
 - Change the detection behavior from immediate conflict detection to waiting when an optimistic transaction meets a pessimistic lock to avoid worsening the lock conflict [#11051](#)

14.9.20.2 TiKV

- Add the statistics of the size of blob files in statistics information [#5060](#)
- Fix the core dump issue caused by the incorrectly cleaned memory resources when the process exits [#5053](#)
- Add all monitoring metrics related to the Titan engine [#4772](#), [#4836](#)
- Add the number of open file handles for Titan when counting the number of open file handles to avoid the issue that no file handle is available because of inaccurate statistics of file handles [#5026](#)
- Set blob_run_mode to decide whether to enable the Titan engine on a specific CF [#4991](#)
- Fix the issue that the read operations cannot get the commit information of pessimistic transactions [#5067](#)

- Add the `blob-run-mode` configuration parameter to control the running mode of the Titan engine, and its value can be `normal`, `read-only` or `fallback` [#4865](#)
- Improve the performance of detecting deadlocks [#5089](#)

14.9.20.3 PD

- Fix the issue that the scheduling limit is automatically adjusted to 0 when PD schedules hot Regions [#1552](#)
- Add the `enable-grpc-gateway` configuration option to enable the gRPC gateway feature of etcd [#1596](#)
- Add `store-balance-rate`, `hot-region-schedule-limit` and other statistics related to scheduler configuration [#1601](#)
- Optimize the hot Region scheduling strategy and skip the Regions that lack replicas during scheduling to prevent multiple replicas from being scheduled to the same IDC [#1609](#)
- Optimize the Region merge processing logic and support giving priority to merging the Regions with smaller sizes to speed up Region merging [#1613](#)
- Adjust the default limit of hot Region scheduling in a single time to 64 to prevent too many scheduling tasks from occupying system resources and impacting performance [#1616](#)
- Optimize the Region scheduling strategy and support giving high priority to scheduling Regions in the `Pending` status [#1617](#)
- Fix the issue that `random-merge` and `admin-merge-region` operators cannot be added [#1634](#)
- Adjust the format of the Region key in the log to hexadecimal notation to make it easier to view [#1639](#)

14.9.20.4 Tools

TiDB Binlog

- Optimize the Pump GC strategy and remove the restriction that the unconsumed binlog cannot be cleaned to make sure that the resources are not occupied for a long time [#646](#)

TiDB Lightning

- Fix the import error that happens when the column names specified by the SQL dump are not in lowercase [#210](#)

14.9.20.5 TiDB Ansible

- Add the precheck feature for the ansible command and its `jmespath` and `jinja2` dependency packages [#803](#), [#813](#)

- Add the `stop-write-at-available-space` parameter (10 GiB by default) in Pump to stop writing binlog files in Pump when the available disk space is less than the parameter value [#806](#)
- Update the I/O monitoring items in the TiKV monitoring information and make them compatible with the monitoring components of the new version [#820](#)
- Update the PD monitoring information, and fix the anomaly that Disk Latency is empty in the disk performance dashboard [#817](#)
- Add monitoring items for Titan in the TiKV details dashboard [#824](#)

14.9.21 TiDB 3.0 GA Release Notes

Release date: June 28, 2019

TiDB version: 3.0.0

TiDB Ansible version: 3.0.0

14.9.21.1 Overview

On June 28, 2019, TiDB 3.0 GA is released. The corresponding TiDB Ansible version is 3.0.0. Compared with TiDB 2.1, this release has greatly improved in the following aspects:

- Stability. TiDB 3.0 has demonstrated long-term stability for large-scale clusters with up to 150+ nodes and 300+ TB of storage.
- Usability. TiDB 3.0 has multi-facet improvements in usability, including standardized slow query logs, well-developed log file specification, and new features such as `EXPLAIN ↗ ANALYZE` and SQL Trace to save operation costs for users.
- Performance. The performance of TiDB 3.0 is 4.5 times greater than TiDB 2.1 in TPC-C benchmarks, and over 1.5 times in Sysbench benchmarks. Thanks to the support for Views, TPC-H 50G Q15 can now run normally.
- New features including Window Functions, Views (**Experimental**), partitioned tables, the plugin framework, pessimistic locking (**Experimental**), and SQL Plan ↗ Management.

14.9.21.2 TiDB

- New Features
 - Support Window Functions; compatible with all window functions in MySQL 8.0, including `NTILE`, `LEAD`, `LAG`, `PERCENT_RANK`, `NTH_VALUE`, `CUME_DIST`, `FIRST_VALUE`, `LAST_VALUE`, `RANK`, `DENSE_RANK`, and `ROW_NUMBER`
 - Support Views (**Experimental**)
 - Improve Table Partition
 - * Support Range Partition
 - * Support Hash Partition

- Add the plug-in framework, supporting plugins such as IP Whitelist (**Enterprise**) and Audit Log (**Enterprise**).
- Support the SQL Plan Management function to create SQL execution plan binding to ensure query stability (**Experimental**)
- SQL Optimizer
 - Optimize the `NOT EXISTS` subquery and convert it to `Anti Semi Join` to improve performance
 - Optimize the constant propagation on the `Outer Join`, and add the optimization rule of `Outer Join` elimination to reduce non-effective computations and improve performance
 - Optimize the `IN` subquery to execute `Inner Join` after aggregation to improve performance
 - Optimize `Index Join` to adapt to more scenarios
 - Improve the Partition Pruning optimization rule of Range Partition
 - Optimize the query logic for `_tidb_rowid` to avoid full table scan and improve performance
 - Match more prefix columns of the indexes when extracting access conditions of composite indexes if there are relevant columns in the filter to improve performance
 - Improve the accuracy of cost estimates by using order correlation between columns
 - Optimize `Join Order` based on the greedy strategy and the dynamic programming algorithm to speed up the join operation of multiple tables
 - Support Skyline Pruning, with some rules to prevent the execution plan from relying too heavily on statistics to improve query stability
 - Improve the accuracy of row count estimation for single-column indexes with `NULL` values
 - Support `FAST ANALYZE` that randomly samples in each Region to avoid full table scan and improve performance with statistics collection
 - Support the incremental Analyze operation on monotonically increasing index columns to improve performance with statistics collection
 - Support using subqueries in the `DO` statement
 - Support using `Index Join` in transactions
 - Optimize `prepare/execute` to support DDL statements with no parameters
 - Modify the system behavior to auto load statistics when the `stats-lease` variable value is 0
 - Support exporting historical statistics
 - Support the `dump/load` correlation of histograms
- SQL Execution Engine
 - Optimize log output: `EXECUTE` outputs user variables and `COMMIT` outputs slow query logs to facilitate troubleshooting
 - Support the `EXPLAIN ANALYZE` function to improve SQL tuning usability
 - Support the `admin show next_row_id` command to get the ID of the next row

- Add six built-in functions: `JSON_QUOTE`, `JSON_ARRAY_APPEND`, `JSON_MERGE_PRESERVE` → , `BENCHMARK`, `COALESCE`, and `NAME_CONST`
- Optimize control logics on the chunk size to dynamically adjust based on the query context, to reduce the SQL execution time and resource consumption
- Support tracking and controlling memory usage in three operators - `TableReader`, `IndexReader` and `IndexLookupReader`
- Optimize the Merge Join operator to support an empty `ON` condition
- Optimize write performance for single tables that contains too many columns
- Improve the performance of `admin show ddl jobs` by supporting scanning data in reverse order
- Add the `split table region` statement to manually split the table Region to alleviate hotspot issues
- Add the `split index region` statement to manually split the index Region to alleviate hotspot issues
- Add a blocklist to prohibit pushing down expressions to Coprocessor
- Optimize the `Expensive Query` log to print the SQL query in the log when it exceeds the configured limit of execution time or memory
- DDL
 - Support migrating from character set `utf8` to `utf8mb4`
 - Change the default character set from `utf8` to `utf8mb4`
 - Add the `alter schema` statement to modify the character set and the collation of the database
 - Support `ALTER` algorithm `INPLACE/INSTANT`
 - Support `SHOW CREATE VIEW`
 - Support `SHOW CREATE USER`
 - Support fast recovery of mistakenly deleted tables
 - Support adjusting the number of concurrencies of `ADD INDEX` dynamically
 - Add the `pre_split_regions` option that pre-allocates Regions when creating the table using the `CREATE TABLE` statement, to relieve write hot Regions caused by lots of writes after the table creation
 - Support splitting Regions by the index and range of the table specified using SQL statements to relieve hotspot issues
 - Add the `ddl_error_count_limit` global variable to limit the number of DDL task retries
 - Add a feature to use `SHARD_ROW_ID_BITS` to scatter row IDs when the column contains an `AUTO_INCREMENT` attribute to relieve hotspot issues
 - Optimize the lifetime of invalid DDL metadata to speed up recovering the normal execution of DDL operations after upgrading the TiDB cluster
- Transactions
 - Support the pessimistic transaction mode (**Experimental**)
 - Optimize transaction processing logics to adapt to more scenarios:
 - * Change the default value `tidb_disable_txn_auto_retry` to `on`, which means non-auto committed transactions will not be retried

- * Add the `tidb_batch_commit` system variable to split a transaction into multiple ones to be executed concurrently
- * Add the `tidb_low_resolution_tso` system variable to control the number of TSOs to obtain in batches and reduce the number of times that transactions request for TSOs, to improve performance in scenarios with relatively low requirement of consistency
- * Add the `tidb_skip_isolation_level_check` variable to control whether to report errors when the isolation level is set to `SERIALIZABLE`
- * Modify the `tidb_disable_txn_auto_retry` system variable to make it work on all retryable errors
- Permission Management
 - Perform permission check on the `ANALYZE`, `USE`, `SET GLOBAL`, and `SHOW PROCESSLIST` statements
 - Support Role Based Access Control (RBAC) (**Experimental**)
- Server
 - Optimize slow query logs:
 - * Restructure the log format
 - * Optimize the log content
 - * Optimize the log query method to support using the `INFORMATION_SCHEMA .SLOW_QUERY` and `ADMIN SHOW SLOW` statements of the memory table to query slow query logs
 - Develop a unified log format specification with restructured log system to facilitate collection and analysis by tools
 - Support using SQL statements to manage TiDB Binlog services, including querying status, enabling TiDB Binlog, maintaining and sending TiDB Binlog strategies.
 - Support using `unix_socket` to connect to the database
 - Support `Trace` for SQL statements
 - Support getting information for a TiDB instance via the `/debug/zip` HTTP interface to facilitate troubleshooting.
 - Optimize monitoring items to facilitate troubleshooting:
 - * Add the `high_error_rate_feedback_total` monitoring item to monitor the difference between the actual data volume and the estimated data volume based on statistics
 - * Add a QPS monitoring item in the database dimension
 - Optimize the system initialization process to only allow the DDL owner to perform the initialization. This reduces the startup time for initialization or upgrading.
 - Optimize the execution logic of `kill query` to improve performance and ensure resource is release properly
 - Add a startup option `config-check` to check the validity of the configuration file
 - Add the `tidb_back_off_weight` system variable to control the backoff time of internal error retries

- Add the `wait_timeout` and `interactive_timeout` system variables to control the maximum idle connections allowed
- Add the connection pool for TiKV to shorten the connection establishing time
- Compatibility
 - Support the `ALLOW_INVALID_DATES` SQL mode
 - Support the MySQL 320 Handshake protocol
 - Support manifesting unsigned BIGINT columns as auto-increment columns
 - Support the `SHOW CREATE DATABASE IF NOT EXISTS` syntax
 - Optimize the fault tolerance of `load data` for CSV files
 - Abandon the predicate pushdown operation when the filtering condition contains a user variable to improve the compatibility with MySQL's behavior of using user variables to simulate Window Functions

14.9.21.3 PD

- Support re-creating a cluster from a single node
- Migrate Region metadata from etcd to the go-leveldb storage engine to solve the storage bottleneck in etcd for large-scale clusters
- API
 - Add the `remove-tombstone` API to clear Tombstone stores
 - Add the `ScanRegions` API to batch query Region information
 - Add the `GetOperator` API to query running operators
 - Optimize the performance of the `GetStores` API
- Configurations
 - Optimize configuration check logic to avoid configuration item errors
 - Add `enable-two-way-merge` to control the direction of Region merge
 - Add `hot-region-schedule-limit` to control the scheduling rate for hot Regions
 - Add `hot-region-cache-hits-threshold` to identify hotspot when hitting multiple thresholds consecutively
 - Add the `store-balance-rate` configuration item to control the maximum numbers of balance Region operators allowed per minute
- Scheduler Optimizations
 - Add the store limit mechanism for separately controlling the speed of operators for each store
 - Support the `waitingOperator` queue to optimize the resource race among different schedulers
 - Support scheduling rate limit to actively send scheduling operations to TiKV. This improves the scheduling rate by limiting the number of concurrent scheduling tasks on a single node.
 - Optimize the `Region Scatter` scheduling to be not restrained by the limit mechanism

- Add the `shuffle-hot-region` scheduler to facilitate TiKV stability test in scenarios of poor hotspot scheduling
- Simulator
 - Add simulator for data import scenarios
 - Support setting different heartbeats intervals for the Store
- Others
 - Upgrade etcd to solve the issues of inconsistent log output formats, Leader selection failure in prevote, and lease deadlocking
 - Develop a unified log format specification with restructured log system to facilitate collection and analysis by tools
 - Add monitoring metrics including scheduling parameters, cluster label information, time consumed by PD to process TSO requests, Store ID and address information, etc.

14.9.21.4 TiKV

- Support distributed GC and concurrent lock resolving for improved GC performance
- Support reversed `raw_scan` and `raw_batch_scan`
- Support Multi-thread Raftstore and Multi-thread Apply to improve scalabilities, concurrency capacity, and resource usage within a single node. Performance improves by 70% under the same level of pressure
- Support batch receiving and sending Raft messages, improving TPS by 7% for write intensive scenarios
- Support checking RocksDB Level 0 files before applying snapshots to avoid write stall
- Introduce Titan, a key-value plugin that improves write performance for scenarios with value sizes greater than 1KiB, and relieves write amplification in certain degrees
- Support the pessimistic transaction mode (**Experimental**)
- Support getting monitoring information via HTTP
- Modify the semantics of `Insert` to allow Prewrite to succeed only when there is no Key
- Develop a unified log format specification with restructured log system to facilitate collection and analysis by tools
- Add performance metrics related to configuration information and key bound crossing
- Support Local Reader in RawKV to improve performance
- Engine
 - Optimize memory management to reduce memory allocation and copying for `Iterator Key Bound Option`
 - Support `block cache` sharing among different column families
- Server
 - Reduce context switch overhead from `batch commands`

- Remove `txn scheduler`
- Add monitoring items related to `read index` and `GC worker`
- RaftStore
 - Support Hibernate Regions to optimize CPU consumption from RaftStore (**Experimental**)
 - Remove the local reader thread
- Coprocessor
 - Refactor the computation framework to implement vector operators, computation using vector expressions, and vector aggregations to improve performance
 - Support providing operator execution status for the `EXPLAIN ANALYZE` statement in TiDB
 - Switch to the `work-stealing` thread pool model to reduce context switch cost

14.9.21.5 Tools

- TiDB Lightning
 - Support redirected replication of data tables
 - Support importing CSV files
 - Improve performance for conversion from SQL to KV pairs
 - Support batch import of single tables to improve performance
 - Support separately importing data and indexes for big tables to improve the performance of TiKV-importer
 - Support filling the missing column using the `row_id` or the default column value when column data is missing in the new file
 - Support setting a speed limit in `TIKV-importer` when uploading SST files to TiKV
- TiDB Binlog
 - Add the `advertise-addr` configuration in Drainer to support the bridge mode in the container environment
 - Add the `GetMvccByEncodeKey` function in Pump to speed up querying the transaction status
 - Support compressing communication data among components to reduce network resource consumption
 - Add the Arbiter tool that supports reading binlog from Kafka and replicate the data into MySQL
 - Support filtering out files that don't require replication via Reparo
 - Support replicating generated columns
 - Add the `syncer.sql-mode` configuration item to support using different sql-modes to parse DDL queries
 - Add the `syncer.ignore-table` configuration item to support filtering tables not to be replicated

- sync-diff-inspector
 - Support checkpoint to record verification status and continue the verification from last saved point after restarting
 - Add the `only-use-checksum` configuration item to check data consistency by calculating checksum
 - Support using TiDB statistics and multiple columns to split chunks for comparison to adapt to more scenarios

14.9.21.6 TiDB Ansible

- Upgrade the following monitoring components to a stable version:
 - Prometheus from V2.2.1 to V2.8.1
 - Pushgateway from V0.4.0 to V0.7.0
 - Node_exporter from V0.15.2 to V0.17.0
 - Alertmanager from V0.14.0 to V0.17.0
 - Grafana from V4.6.3 to V6.1.6
 - Ansible from V2.5.14 to V2.7.11
- Add the TiKV summary monitoring dashboard to view cluster status conveniently
- Add the TiKV trouble_shooting monitoring dashboard to remove duplicate items and facilitate troubleshooting
- Add the TiKV details monitoring dashboard to facilitate debugging and troubleshooting
- Add concurrent check for version consistency during rolling updates to improve the update performance
- Support deployment and operations for TiDB Lightning
- Optimize the `table-regions.py` script to support displaying Leader distribution by tables
- Optimize TiDB monitoring and add latency related monitoring items by SQL categories
- Modify the operating system version limit to only support the CentOS 7.0+ and Red Hat 7.0+ operating systems
- Add the monitoring item to predict the maximum QPS of the cluster (hidden by default)

14.9.22 TiDB 3.0.0-rc.3 Release Notes

Release date: June 21, 2019

TiDB version: 3.0.0-rc.3

TiDB Ansible version: 3.0.0-rc.3

14.9.22.1 Overview

On June 21, 2019, TiDB 3.0.0-rc.3 is released. The corresponding TiDB Ansible version is 3.0.0-rc.3. Compared with TiDB 3.0.0-rc.2, this release has greatly improved the stability, usability, features, the SQL optimizer, statistics, and the execution engine.

14.9.22.2 TiDB

- SQL Optimizer
 - Remove the feature of collecting virtual generated column statistics [#10629](#)
 - Fix the issue that the primary key constant overflows during point queries [#10699](#)
 - Fix the issue that using uninitialized information in `fast analyze` causes panic [#10691](#)
 - Fix the issue that executing the `create view` statement using `prepare` causes panic because of wrong column information [#10713](#)
 - Fix the issue that the column information is not cloned when handling window functions [#10720](#)
 - Fix the wrong estimation for the selectivity rate of the inner table selection in index join [#10854](#)
 - Support automatic loading statistics when the `stats-lease` variable value is 0 [#10811](#)
- Execution Engine
 - Fix the issue that resources are not correctly released when calling the `Close` function in `StreamAggExec` [#10636](#)
 - Fix the issue that the order of `table_option` and `partition_options` is incorrect in the result of executing the `show create table` statement for partitioned tables [#10689](#)
 - Improve the performance of `admin show ddl jobs` by supporting scanning data in reverse order [#10687](#)
 - Fix the issue that the result of the `show grants` statement in RBAC is incompatible with that of MySQL when this statement has the `current_user` field [#10684](#)
 - Fix the issue that UUIDs might generate duplicate values on multiple nodes [#10712](#)
 - Fix the issue that the `show view` privilege is not considered in `explain` [#10635](#)
 - Add the `split table region` statement to manually split the table Region to alleviate the hotspot issue [#10765](#)
 - Add the `split index region` statement to manually split the index Region to alleviate the hotspot issue [#10764](#)
 - Fix the incorrect execution issue when you execute multiple statements such as `create user`, `grant`, or `revoke` consecutively [#10737](#)
 - Add a blocklist to prohibit pushing down expressions to Coprocessor [#10791](#)
 - Add the feature of printing the `expensive query` log when a query exceeds the memory configuration limit [#10849](#)

- Add the `bind-info-lease` configuration item to control the update time of the modified binding execution plan [#10727](#)
- Fix the OOM issue in high concurrent scenarios caused by the failure to quickly release Coprocessor resources, resulted from the `execdetails.ExecDetails` pointer [#10832](#)
- Fix the panic issue caused by the `kill` statement in some cases [#10876](#)
- Server
 - Fix the issue that goroutine might leak when repairing GC [#10683](#)
 - Support displaying the `host` information in slow queries [#10693](#)
 - Support reusing idle links that interact with TiKV [#10632](#)
 - Fix the support for enabling the `skip-grant-table` option in RBAC [#10738](#)
 - Fix the issue that `pessimistic-txn` configuration goes invalid [#10825](#)
 - Fix the issue that the actively canceled tictclient requests are still retried [#10850](#)
 - Improve performance in the case where pessimistic transactions conflict with optimistic transactions [#10881](#)
- DDL
 - Fix the issue that modifying charset using `alter table` causes the `blob` type change [#10698](#)
 - Add a feature to use `SHARD_ROW_ID_BITS` to scatter row IDs when the column contains an `AUTO_INCREMENT` attribute to alleviate the hotspot issue [#10794](#)
 - Prohibit adding stored generated columns by using the `alter table` statement [#10808](#)
 - Optimize the invalid survival time of DDL metadata to shorten the period during which the DDL operation is slower after cluster upgrade [#10795](#)

14.9.22.3 PD

- Add the `enable-two-way-merge` configuration item to allow only one-way merging [#1583](#)
- Add scheduling operations for `AddLightLearner` and `AddLightPeer` to make Region Scatter scheduling unrestricted by the limit mechanism [#1563](#)
- Fix the issue of insufficient reliability because the data might only have one replica replication when the system is started [#1581](#)
- Optimize configuration check logic to avoid configuration item errors [#1585](#)
- Adjust the definition of the `store-balance-rate` configuration to the upper limit of the number of balance operators generated per minute [#1591](#)
- Fix the issue that the store might have been unable to generate scheduled operations [#1590](#)

14.9.22.4 TiKV

- Engine

- Fix the issue that incomplete snapshots are generated in the system caused by the iterator not checking the status [#4936](#)
- Fix the data loss issue caused by a delay of flushing data to the disk when receiving snapshots after a power failure in abnormal conditions [#4850](#)
- Server
 - Add a feature to check the validity of the `block-size` configuration [#4928](#)
 - Add `READ_INDEX`-related monitoring metrics [#4830](#)
 - Add GC worker-related monitoring metrics [#4922](#)
- Raftstore
 - Fix the issue that the cache of the local reader is not cleared correctly [#4778](#)
 - Fix the issue that the request delay might be increased when transferring the leader and changing conf [#4734](#)
 - Fix the issue that a stale command is wrongly reported [#4682](#)
 - Fix the issue that the command might be pending for a long time [#4810](#)
 - Fix the issue that files are damaged after a power failure, which is caused by a delay of synchronizing the snapshot file to the disk [#4807](#), [#4850](#)
- Coprocessor
 - Support Top-N in vector calculation [#4827](#)
 - Support `Stream` aggregation in vector calculation [#4786](#)
 - Support the `AVG` aggregate function in vector calculation [#4777](#)
 - Support the `First` aggregate function in vector calculation [#4771](#)
 - Support the `SUM` aggregate function in vector calculation [#4797](#)
 - Support the `MAX/MIN` aggregate function in vector calculation [#4837](#)
 - Support the `Like` expression in vector calculation [#4747](#)
 - Support the `MultiplyDecimal` expression in vector calculation [#4849](#)
 - Support the `BitAnd/BitOr/BitXor` expression in vector calculation [#4724](#)
 - Support the `UnaryNot` expression in vector calculation [#4808](#)
- Transaction
 - Fix the issue that an error occurs caused by non-pessimistic locking conflicts in pessimistic transactions [#4801](#), [#4883](#)
 - Reduce unnecessary calculation for optimistic transactions after enabling pessimistic transactions to improve the performance [#4813](#)
 - Add a feature of single statement rollback to ensure that the whole transaction does not need a rollback operation in a deadlock situation [#4848](#)
 - Add pessimistic transaction-related monitoring items [#4852](#)
 - Support using the `ResolveLockLite` command to resolve lightweight locks to improve the performance when severe conflicts exist [#4882](#)
- tikv-ctl
 - Add the `bad-regions` command to support checking more abnormal conditions [#4862](#)

- Add a feature of forcibly executing the `tombstone` command [#4862](#)
- Misc
 - Add the `dist_release` compiling command [#4841](#)

14.9.22.5 Tools

- TiDB Binlog
 - Fix the wrong offset issue caused by Pump not checking the returned value when it fails to write data [#640](#)
 - Add the `advertise-addr` configuration in Drainer to support the bridge mode in the container environment [#634](#)
 - Add the `GetMvccByEncodeKey` function in Pump to speed up querying the transaction status [#632](#)

14.9.22.6 TiDB Ansible

- Add a monitoring item to predict the maximum QPS value of the cluster (“hide” by default) [#f5cfa4d](#)

14.9.23 TiDB 3.0.0-rc.2 Release Notes

Release date: May 28, 2019

TiDB version: 3.0.0-rc.2

TiDB Ansible version: 3.0.0-rc.2

14.9.23.1 Overview

On May 28, 2019, TiDB 3.0.0-rc.2 is released. The corresponding TiDB Ansible version is 3.0.0-rc.2. Compared with TiDB 3.0.0-rc.1, this release has greatly improved the stability, usability, features, the SQL optimizer, statistics, and the execution engine.

14.9.23.2 TiDB

- SQL Optimizer
 - Support Index Join in more scenarios [#10540](#)
 - Support exporting historical statistics [#10291](#)
 - Support the incremental `Analyze` operation on monotonically increasing index columns [#10355](#)
 - Neglect the NULL value in the `Order By` clause [#10488](#)

- Fix the wrong schema information calculation of the `UnionAll` logical operator when simplifying the column information [#10384](#)
- Avoid modifying the original expression when pushing down the `Not` operator [#10363](#)
- Support the `dump/load` correlation of histograms [#10573](#)
- Execution Engine
 - Handle virtual columns with a unique index properly when fetching duplicate rows in `batchChecker` [#10370](#)
 - Fix the scanning range calculation issue for the `CHAR` column [#10124](#)
 - Fix the issue of `PointGet` incorrectly processing negative numbers [#10113](#)
 - Merge `Window` functions with the same name to improve execution efficiency [#9866](#)
 - Allow the `RANGE` frame in a `Window` function to contain no `OrderBy` clause [#10496](#)
- Server
 - Fix the issue that TiDB continuously creates a new connection to TiKV when a fault occurs in TiKV [#10301](#)
 - Make `tidb_disable_txn_auto_retry` affect all retryable errors instead of only write conflict errors [#10339](#)
 - Allow DDL statements without parameters to be executed using `prepare ↪ /execute` [#10144](#)
 - Add the `tidb_back_off_weight` variable to control the backoff time [#10266](#)
 - Prohibit TiDB retrying non-automatically committed transactions in default conditions by setting the default value of `tidb_disable_txn_auto_retry` to on [#10266](#)
 - Fix the database privilege judgment of `role` in RBAC [#10261](#)
 - Support the pessimistic transaction mode (experimental) [#10297](#)
 - Reduce the wait time for handling lock conflicts in some cases [#10006](#)
 - Make the Region cache able to visit follower nodes when a fault occurs in the leader node [#10256](#)
 - Add the `tidb_low_resolution_tso` variable to control the number of TSOs obtained in batches and reduce the times of transactions obtaining TSO to adapt for scenarios where data consistency is not so strictly required [#10428](#)
- DDL
 - Fix the uppercase issue of the charset name in the storage of the old version of TiDB [#10272](#)
 - Support `preSplit` of table partition, which pre-allocates table Regions when creating a table to avoid write hotspots after the table is created [#10221](#)
 - Fix the issue that TiDB incorrectly updates the version information in PD in some cases [#10324](#)
 - Support modifying the charset and collation using the `ALTER DATABASE` statement [#10393](#)

- Support splitting Regions based on the index and range of the specified table to relieve hotspot issues [#10203](#)
- Prohibit modifying the precision of the decimal column using the `alter table` statement [#10433](#)
- Fix the restriction for expressions and functions in hash partition [#10273](#)
- Fix the issue that adding indexes in a table that contains partitions will in some cases cause TiDB panic [#10475](#)
- Validate table information before executing the DDL to avoid invalid table schemas [#10464](#)
- Enable hash partition by default; and enable range columns partition when there is only one column in the partition definition [#9936](#)

14.9.23.3 PD

- Enable the Region storage by default to store the Region metadata [#1524](#)
- Fix the issue that hot Region scheduling is preempted by another scheduler [#1522](#)
- Fix the issue that the priority for the leader does not take effect [#1533](#)
- Add the gRPC interface for `ScanRegions` [#1535](#)
- Push operators actively [#1536](#)
- Add the store limit mechanism for separately controlling the speed of operators for each store [#1474](#)
- Fix the issue of inconsistent Config status [#1476](#)

14.9.23.4 TiKV

- Engine
 - Support multiple column families sharing a block cache [#4563](#)
- Server
 - Remove `TxnScheduler` [#4098](#)
 - Support pessimistic lock transactions [#4698](#)
- Raftstore
 - Support hibernate Regions to reduce the consumption of the raftstore CPU [#4591](#)
 - Fix the issue that the leader does not reply to the `ReadIndex` requests for the learner [#4653](#)
 - Fix transferring leader failures in some cases [#4684](#)
 - Fix the dirty read issue in some cases [#4688](#)
 - Fix the issue that a snapshot may lose applied data in some cases [#4716](#)
- Coprocessor
 - Add more RPN functions
 - * `LogicalOr` [#4691](#)

- * `LTReal` #4602
- * `LEReal` #4602
- * `GTReal` #4602
- * `GEReal` #4602
- * `NEReal` #4602
- * `EQReal` #4602
- * `IsNull` #4720
- * `IsTrue` #4720
- * `IsFalse` #4720
- * Support comparison arithmetic for `Int` #4625
- * Support comparison arithmetic for `Decimal` #4625
- * Support comparison arithmetic for `String` #4625
- * Support comparison arithmetic for `Time` #4625
- * Support comparison arithmetic for `Duration` #4625
- * Support comparison arithmetic for `Json` #4625
- * Support plus arithmetic for `Int` #4733
- * Support plus arithmetic for `Real` #4733
- * Support plus arithmetic for `Decimal` #4733
- * Support MOD functions for `Int` #4727
- * Support MOD functions for `Real` #4727
- * Support MOD functions for `Decimal` #4727
- * Support minus arithmetic for `Int` #4746
- * Support minus arithmetic for `Real` #4746
- * Support minus arithmetic for `Decimal` #4746

14.9.23.5 Tools

- TiDB Binlog
 - Add a metric to track the delay of data replication downstream #594
- TiDB Lightning
 - Support merging sharded databases and tables #95
 - Add the retry mechanism for KV write failure #176
 - Update the default value of `table-concurrency` to 6 #175
 - Reduce required configuration items by automatically discovering `tidb.pd-addr` and `tidb.port` if they are not provided #173

14.9.24 TiDB 3.0.0-rc.1 Release Notes

Release Date: May 10, 2019

TiDB version: 3.0.0-rc.1

TiDB Ansible version: 3.0.0-rc.1

14.9.24.1 Overview

On May 10, 2019, TiDB 3.0.0-rc.1 is released. The corresponding TiDB Ansible version is 3.0.0-rc.1. Compared with TiDB 3.0.0-beta.1, this release has greatly improved the stability, usability, features, the SQL optimizer, statistics, and the execution engine.

14.9.24.2 TiDB

- SQL Optimizer
 - Improve the accuracy of cost estimates by using order correlation between columns; introduce a heuristic parameter `tidb_opt_correlation_exp_factor` to control the preference for index scans for scenarios when correlation cannot be directly used for estimation. [#9839](#)
 - Match more prefix columns of the indexes when extracting access conditions of composite indexes if there are relevant columns in the filter [#10053](#)
 - Use the dynamic programming algorithm to specify the execution order of join operations when the number of tables participating in the join is less than the value of `tidb_opt_join_reorder_threshold`. [#8816](#)
 - Match more prefix columns of the indexes in the inner tables that build the index join when using composite indexes as the access conditions [#8471](#)
 - Improve the accuracy of row count estimation for single-column indexes with NULL values [#9474](#)
 - Specially handle `GROUP_CONCAT` when eliminating aggregate functions during the logical optimization phase to prevent incorrect executions [#9967](#)
 - Properly push the filter down to child nodes of the join operator if the filter is a constant [#9848](#)
 - Specially handle some functions such as `RAND()` when pruning columns during the logical optimization phase to prevent incompatibilities with MySQL [#10064](#)
 - Support `FAST ANALYZE`, which speeds up statistics collection by sampling the region instead of scanning the entire region. This feature is controlled by the variable `tidb_enable_fast_analyze`. [#10258](#)
 - Support SQL Plan Management, which ensures execution stability by performing execution plan binding for SQL statements. This feature is currently in beta and only supports bound execution plans for `SELECT` statements. It is not recommended to use it in the production environment. [#10284](#)
- Execution Engine
 - Support tracking and controlling memory usage in three operators - `TableReader`, `IndexReader` and `IndexLookupReader` [#10003](#)
 - Support showing more information about coprocessor tasks in the slow log such as the number of tasks in coprocessor, the average/longest/90% of execution/waiting time and the addresses of the TiKVs which take the longest execution time or waiting time [#10165](#)
 - Support the prepared DDL statements with no placeholders [#10144](#)

- Server

- Only allow the DDL owner to execute bootstrap when TiDB is started [#10029](#)
- Add the variable `tidb_skip_isolation_level_check` to prevent TiDB from reporting errors when setting the transaction isolation level to SERIALIZABLE [#10065](#)
- Merge the implicit commit time and the SQL execution time in the slow log [#10294](#)
 - * Support for SQL Roles (RBAC Privilege Management)
 - * Support `SHOW GRANT` [#10016](#)
 - * Support `SET DEFAULT ROLE` [#9949](#)
- Support `GRANT ROLE` [#9721](#)
- Fix the `ConnectionEvent` error from the `whitelist` plugin that makes TiDB exit [#9889](#)
- Fix the issue of mistakenly adding read-only statements to the transaction history [#9723](#)
- Improve `kill` statements to stop SQL execution and release resources more quickly [#9844](#)
- Add a startup option `config-check` to check the validity of the configuration file [#9855](#)
- Fix the validity check of inserting NULL fields when the strict SQL mode is disabled [#10161](#)

- DDL

- Add the `pre_split_regions` option for `CREATE TABLE` statements; this option supports pre-splitting the Table Region when creating a table to avoid write hot spots caused by lots of writes after the table creation [#10138](#)
- Optimize the execution performance of some DDL statements [#10170](#)
- Add the warning that full-text indexes are not supported for `FULLTEXT KEY` [#9821](#)
- Fix the compatibility issue for the UTF8 and UTF8MB4 charsets in the old versions of TiDB [#9820](#)
- Fix the potential bug in `shard_row_id_bits` of a table [#9868](#)
- Fix the bug that the column charset is not changed after the table charset is changed [#9790](#)
- Fix a potential bug in `SHOW COLUMN` when using `BINARY/BIT` as the column default value [#9897](#)
- Fix the compatibility issue in displaying `CHARSET/COLLATION` descriptions in the `SHOW FULL COLUMNS` statement [#10007](#)
- Fix the issue that the `SHOW COLLATIONS` statement only lists collations supported by TiDB [#10186](#)

14.9.24.3 PD

- Upgrade ETCD [#1452](#)

- Unify the log format of etcd and PD server
- Fix the issue of failing to elect Leader by PreVote
- Support fast dropping the “propose” and “read” requests that are to fail to avoid blocking the subsequent requests
- Fix the deadlock issue of Lease
- Fix the issue that a hot store makes incorrect statistics of keys [#1487](#)
- Support forcibly rebuilding a PD cluster from a single PD node [#1485](#)
- Fix the issue that `regionScatterer` might generate an invalid `OperatorStep` [#1482](#)
- Fix the too short timeout issue of the `MergeRegion` operator [#1495](#)
- Support giving high priority to hot region scheduling [#1492](#)
- Add the metrics for recording the time of handling TSO requests on the PD server side [#1502](#)
- Add the corresponding Store ID and Address to the metrics related to the store [#1506](#)
- Support the `GetOperator` service [#1477](#)
- Fix the issue that the error cannot be sent in the Heartbeat stream because the store cannot be found [#1521](#)

14.9.24.4 TiKV

- Engine
 - Fix the issue that may cause incorrect statistics on read traffic [#4436](#)
 - Fix the issue that may cause prefix extractor panic when deleting a range [#4503](#)
 - Optimize memory management to reduce memory allocation and copying for `Iterator Key Bound Option` [#4537](#)
 - Fix the issue that failing to consider learner log gap may in some cases cause panic [#4559](#)
 - Support `block cache` sharing among different `column families` [#4612](#)
- Server
 - Reduce context switch overhead of `batch commands` [#4473](#)
 - Check the validity of seek iterator status [#4470](#)
- RaftStore
 - Support configurable `properties index distance` [#4517](#)
- Coprocessor
 - Add batch index scan executor [#4419](#)
 - Add vectorized evaluation framework [#4322](#)
 - Add execution summary framework for batch executors [#4433](#)
 - Check the maximum column when constructing the RPN expression to avoid invalid column offset that may cause evaluation panic [#4481](#)
 - Add `BatchLimitExecutor` [#4469](#)

- Replace the original `futures-cpupool` with `tokio-threadpool` in ReadPool to reduce context switch [#4486](#)
- Add batch aggregation framework [#4533](#)
- Add `BatchSelectionExecutor` [#4562](#)
- Add batch aggression function `AVG` [#4570](#)
- Add RPN function `LogicalAnd`[#4575](#)
- Misc
 - Support `tcmalloc` as a memory allocator [#4370](#)

14.9.24.5 Tools

- TiDB Binlog
 - Fix the replication abortion issue when binlog data for the primary key column of unsigned int type is negative [#573](#)
 - Provide no compression option when downstream is `pb`; modify the downstream name from `pb` to `file` [#559](#)
 - Add the `storage.sync-log` configuration item in Pump that allows asynchronous flush on local storage [#509](#)
 - Support traffic compression for communications between Pump and Drainer [#495](#)
 - Add the `syncer.sql-mode` configuration item in Drainer to support parsing DDL queries in different sql-mode [#511](#)
 - Add the `syncer.ignore-table` configuration item to support filtering out tables that do not require replication [#520](#)
- Lightning
 - Use row IDs or default column values to populate the column data missed in the dump file [#170](#)
 - Fix the bug in Importer that import success may still be returned even if part of the SST failed to be imported [#4566](#)
 - Support speed limit in Importer when uploading SST to TiKV [#4412](#)
 - Support importing tables by size to reduce impacts on the cluster brought by Checksum and Analyze for big tables, and improve the success rate for Checksum and Analyze [#156](#)
 - Improve Lightning’s SQL encoding performance by 50% by directly parsing data source file as types.Datum of TiDB and saving extra parsing overhead from the KV encoder [#145](#)
 - Change log format to [Unified Log Format](#) [#162](#)
 - Add some command line options for use when the configuration file is missing [#157](#)
- sync-diff-inspector
 - Support checkpoint to record verification status and continue the verification from last saved point after restarting [#224](#)

- Add the `only-use-checksum` configuration item to check data consistency by calculating checksum [#215](#)

14.9.24.6 TiDB Ansible

- Support more TiKV monitoring panels and update versions for Ansible, Grafana, and Prometheus [#727](#)
 - Summary dashboard for viewing cluster status
 - trouble_shooting dashboard for troubleshooting issues
 - Details dashboard for developers to analyze issues
- Fix the bug that causes the downloading failure of TiDB Binlog of Kafka version [#730](#)
- Modify version limits on supported operating systems as CentOS 7.0+ and later, and Red Hat 7.0 and later [#733](#)
- Change version detection mode during the rolling update to multi-concurrent [#736](#)
- Update documentation links in README [#740](#)
- Remove redundant TiKV monitoring metrics; add new metrics for troubleshooting [#735](#)
- Optimize `table-regions.py` script to display leader distribution by table [#739](#)
- Update configuration file for Drainer [#745](#)
- Optimize TiDB monitoring with new panels that display latencies by SQL categories [#747](#)
- Update the Lightning configuration file and add the `tidb_lightning_ctl` script [#1e946f8](#)

14.9.25 TiDB 3.0.0 Beta.1 Release Notes

Release Date: March 26, 2019

TiDB version: 3.0.0-beta.1

TiDB Ansible version: 3.0.0-beta.1

14.9.25.1 Overview

On March 26, 2019, TiDB 3.0.0 Beta.1 is released. The corresponding TiDB Ansible version is 3.0.0 Beta.1. Compared with TiDB 3.0.0 Beta, this release has greatly improved the stability, usability, features, the SQL optimizer, statistics, and the execution engine.

14.9.25.2 TiDB

- SQL Optimizer
 - Support calculating the Cartesian product by using Sort Merge Join [#9032](#)

- Support Skyline Pruning, with some rules to prevent the execution plan from relying too heavily on statistics [#9337](#)
- Support Window Functions
 - * `NTILE` [#9682](#)
 - * `LEAD` and `LAG` [#9672](#)
 - * `PERCENT_RANK` [#9671](#)
 - * `NTH_VALUE` [#9596](#)
 - * `CUME_DIST` [#9619](#)
 - * `FIRST_VALUE` and `LAST_VALUE` [#9560](#)
 - * `RANK` and `DENSE_RANK` [#9500](#)
 - * `RANGE FRAMED` [#9450](#)
 - * `ROW FRAMED` [#9358](#)
 - * `ROW NUMBER` [#9098](#)
- Add a type of statistic that indicates the order correlation between columns and the handle column [#9315](#)
- SQL Execution Engine
 - Add built-in functions
 - * `JSON_QUOTE` [#7832](#)
 - * `JSON_ARRAY_APPEND` [#9609](#)
 - * `JSON_MERGE_PRESERVE` [#8931](#)
 - * `BENCHMARK` [#9252](#)
 - * `COALESCE` [#9087](#)
 - * `NAME_CONST` [#9261](#)
 - Optimize the Chunk size based on the query context, to reduce the execution time of SQL statements and resources consumption of the cluster [#6489](#)
- Privilege management
 - Support `SET ROLE` and `CURRENT_ROLE` [#9581](#)
 - Support `DROP ROLE` [#9616](#)
 - Support `CREATE ROLE` [#9461](#)
- Server
 - Add the `/debug/zip` HTTP interface to get information of the current TiDB instance [#9651](#)
 - Support the `show pump status` and `show drainer status` SQL statements to check the Pump or Drainer status [#9456](#)
 - Support modifying the Pump or Drainer status by using SQL statements [#9789](#)
 - Support adding HASH fingerprints to SQL text for easy tracking of slow SQL statements [#9662](#)
 - Add the `log_bin` system variable (“0” by default) to control the enabling state of binlog; only support checking the state currently [#9343](#)
 - Support managing the sending binlog strategy by using the configuration file [#9864](#)

- Support querying the slow log by using the `INFORMATION_SCHEMA.SLOW_QUERY` memory table [#9290](#)
- Change the MySQL version displayed in TiDB from 5.7.10 to 5.7.25 [#9553](#)
- Unify the `log format` for easy collection and analysis by tools
- Add the `high_error_rate_feedback_total` monitoring item to record the difference between the actual data volume and the estimated data volume based on statistics [#9209](#)
- Add the QPS monitoring item in the database dimension, which can be enabled by using a configuration item [#9151](#)

- DDL

- Add the `ddl_error_count_limit` global variable (“512” by default) to limit the number of DDL task retries (If this number exceeds the limit, the DDL task is canceled) [#9295](#)
- Support `ALTER ALGORITHM INPLACE/INSTANT` [#8811](#)
- Support the `SHOW CREATE VIEW` statement [#9309](#)
- Support the `SHOW CREATE USER` statement [#9240](#)

14.9.25.3 PD

- Unify the `log format` for easy collection and analysis by tools
- Simulator
 - Support different heartbeat intervals in different stores [#1418](#)
 - Add a case about importing data [#1263](#)
- Make hotspot scheduling configurable [#1412](#)
- Add the store address as the dimension monitoring item to replace the previous Store ID [#1429](#)
- Optimize the `GetStores` overhead to speed up the Region inspection cycle [#1410](#)
- Add an interface to delete the Tombstone Store [#1472](#)

14.9.25.4 TiKV

- Optimize the Coprocessor calculation execution framework and implement the TableScan section, with the Single TableScan performance improved by 5% ~ 30%
 - Implement the definition of the `BatchRows` row and the `BatchColumn` column [#3660](#)
 - Implement `VectorLike` to support accessing encoded and decoded data in the same way [#4242](#)
 - Define the `BatchExecutor` to interface and implement the way of converting requests to `BatchExecutor` [#4243](#)
 - Implement transforming the expression tree into the RPN format [#4329](#)

- Implement the `BatchTableScanExecutor` vectorization operator to accelerate calculation [#4351](#)
- Unify the `log` format for easy collection and analysis by tools
- Support using the Local Reader to read in the Raw Read interface [#4222](#)
- Add metrics about configuration information [#4206](#)
- Add metrics about key exceeding bound [#4255](#)
- Add an option to control panic or return an error when encountering the key exceeding bound error [#4254](#)
- Add support for the `INSERT` operation, make prewrite succeed only when keys do not exist, and eliminate `Batch Get` [#4085](#)
- Use more fair batch strategy in the Batch System [#4200](#)
- Support Raw scan in `tikv-ctl` [#3825](#)

14.9.25.5 Tools

- TiDB Binlog
 - Add the Arbiter tool that supports reading binlog from Kafka and replicate the data into MySQL
 - Support filtering files that do not need to be replicated
 - Support replicating generated columns
- Lightning
 - Support disabling TiKV periodic Level-1 compaction, and when the TiKV cluster version is 2.1.4 or later, Level-1 compaction is automatically executed in the import mode [#119](#), [#4199](#)
 - Add the `table_concurrency` configuration item to limit the number of import engines (“16” by default) and avoid overusing the importer disk space [#119](#)
 - Support saving the intermediate state SST to the disk, to reduce memory usage [#4369](#)
 - Optimize the import performance of TiKV-Importer and support separate import of data and indexes for large tables [#132](#)
 - Support importing CSV files [#111](#)
- Data replication comparison tool (`sync-diff-inspector`)
 - Support using TiDB statistics to split chunks to be compared [#197](#)
 - Support using multiple columns to split chunks to be compared [#197](#)

14.9.26 TiDB 3.0 Beta Release Notes

On January 19, 2019, TiDB 3.0 Beta is released. The corresponding TiDB Ansible 3.0 Beta is also released. TiDB 3.0 Beta builds on TiDB 2.1 with an added focus in stability, the SQL optimizer, statistics, and the execution engine.

14.9.26.1 TiDB

- New Features
 - Support Views
 - Support Window Functions
 - Support Range Partitioning
 - Support Hash Partitioning
- SQL Optimizer
 - Re-support the optimization rule of AggregationElimination [#7676](#)
 - Optimize the NOT EXISTS subquery and convert it to Anti Semi Join [#7842](#)
 - Add the `tidb_enable_cascades_planner` variable to support the new Cascades optimizer. Currently, the Cascades optimizer is not yet fully implemented and is turned off by default [#7879](#)
 - Support using Index Join in transactions [#7877](#)
 - Optimize the constant propagation on the Outer Join, so that the filtering conditions related to the Outer table in the Join result can be pushed down through the Outer Join to the Outer table, reducing the useless calculation of the Outer Join and improving the execution performance [#7794](#)
 - Adjust the optimization rule of Projection Elimination to the position after the Aggregation Elimination, to avoid redundant Project operators [#7909](#)
 - Optimize the IFNULL function and eliminate this function when the input parameter has a non-NULL attribute [#7924](#)
 - Support Range for `_tidb_rowid` construction queries, to avoid full table scan and reduce cluster stress [#8047](#)
 - Optimize the IN subquery to do the Inner Join after the aggregation, and add the `tidb_opt_insubq_to_join_and_agg` variable to control whether to enable this optimization rule and open it by default [#7531](#)
 - Support using subqueries in the DO statement [#8343](#)
 - Add the optimization rule of Outer Join elimination to reduce unnecessary table scan and Join operations and improve execution performance [#8021](#)
 - Modify the Hint behavior of the TIDB_INLJ optimizer, and the optimizer will use the table specified in Hint as the Inner table of Index Join [#8243](#)
 - Use PointGet in a wide range so that it can be used when the execution plan cache of the Prepare statement takes effect [#8108](#)
 - Introduce the greedy Join Reorder algorithm to optimize the join order selection when joining multiple tables [#8394](#)
 - Support View [#8757](#)
 - Support Window Function [#8630](#)
 - Return warning to the client when TIDB_INLJ is not in effect, to enhance usability [#9037](#)
 - Support deducing the statistics for filtered data based on filtering conditions and table statistics [#7921](#)
 - Improve the Partition Pruning optimization rule of Range Partition [#8885](#)

- SQL Executor
 - Optimize the `Merge Join` operator to support the empty `ON` condition [#9037](#)
 - Optimize the log and print the user variables used when executing the `EXECUTE` statement [#7684](#)
 - Optimize the log to print slow query information for the `COMMIT` statement [#7951](#)
 - Support the `EXPLAIN ANALYZE` feature to make the SQL tuning process easier [#7827](#)
 - Optimize the write performance of wide tables with many columns [#7935](#)
 - Support `admin show next_row_id` [#8242](#)
 - Add the `tidb_init_chunk_size` variable to control the size of the initial Chunk used by the execution engine [#8480](#)
 - Improve `shard_row_id_bits` and cross-check the auto-increment ID [#8936](#)
- Prepare Statement
 - Prohibit adding the `Prepare` statement containing subqueries to the query plan cache to guarantee the query plan is correct when different user variables are input [#8064](#)
 - Optimize the query plan cache to guarantee the plan can be cached when the statement contains non-deterministic functions [#8105](#)
 - Optimize the query plan cache to guarantee the query plan of `DELETE/UPDATE ↪ /INSERT` can be cached [#8107](#)
 - Optimize the query plan cache to remove the corresponding plan when executing the `DEALLOCATE` statement [#8332](#)
 - Optimize the query plan cache to avoid the TiDB OOM issue caused by caching too many plans by limiting the memory usage [#8339](#)
 - Optimize the `Prepare` statement to support using the `?` placeholder in the `ORDER ↪ BY/GROUP BY/LIMIT` clause [#8206](#)
- Privilege Management
 - Add the privilege check for the `ANALYZE` statement [#8486](#)
 - Add the privilege check for the `USE` statement [#8414](#)
 - Add the privilege check for the `SET GLOBAL` statement [#8837](#)
 - Add the privilege check for the `SHOW PROCESSLIST` statement [#7858](#)
- Server
 - Support the `Trace` feature [#9029](#)
 - Support the plugin framework [#8788](#)
 - Support using `unix_socket` and TCP simultaneously to connect to the database [#8836](#)
 - Support the `interactive_timeout` system variable [#8573](#)
 - Support the `wait_timeout` system variable [#8346](#)
 - Support splitting a transaction into multiple transactions based on the number of statements using the `tidb_batch_commit` variable [#8293](#)
 - Support using the `ADMIN SHOW SLOW` statement to check slow logs [#7785](#)

- Compatibility
 - Support the `ALLOW_INVALID_DATES` SQL mode [#9027](#)
 - Improve `LoadData` fault-tolerance for the CSV file [#9005](#)
 - Support the MySQL 320 handshake protocol [#8812](#)
 - Support using the unsigned `bigint` column as the auto-increment column [#8181](#)
 - Support the `SHOW CREATE DATABASE IF NOT EXISTS` syntax [#8926](#)
 - Abandon the predicate pushdown operation when the filtering condition contains a user variable to improve the compatibility with MySQL's behavior of using user variables to mock the Window Function behavior [#8412](#)
- DDL
 - Support fast recovery of mistakenly deleted tables [#7937](#)
 - Support adjusting the number of concurrencies of `ADD INDEX` dynamically [#8295](#)
 - Support changing the character set of tables or columns to `utf8/utf8mb4` [#8037](#)
 - Change the default character set from `utf8` to `utf8mb4` [#7965](#)
 - Support Range Partition [#8011](#)

14.9.26.2 Tools

- TiDB Lightning
 - Speed up converting SQL statements to KV pairs remarkably [#110](#)
 - Support batch import for a single table to improve import performance and stability [#113](#)

14.9.26.3 PD

- Add `RegionStorage` to store Region metadata separately [#1237](#)
- Add shuffle hot Region scheduler [#1361](#)
- Add scheduling parameter related metrics [#1406](#)
- Add cluster label related metrics [#1402](#)
- Add the importing data simulator [#1263](#)
- Fix the `Watch` issue about leader election [#1396](#)

14.9.26.4 TiKV

- Support distributed GC [#3179](#)
- Check RocksDB Level 0 files before applying snapshots to avoid Write Stall [#3606](#)
- Support reverse `raw_scan` and `raw_batch_scan` [#3742](#)
- Support using HTTP to obtain monitoring information [#3855](#)
- Support DST better [#3786](#)
- Support receiving and sending Raft messages in batch [#3931](#)
- Introduce a new storage engine Titan [#3985](#)

- Upgrade gRPC to v1.17.2 [#4023](#)
- Support receiving the client requests and sending replies in batch [#4043](#)
- Support multi-thread Apply [#4044](#)
- Support multi-thread Raftstore [#4066](#)

14.10 v2.1

14.10.1 TiDB 2.1.19 Release Notes

Release date: December 27, 2019

TiDB version: 2.1.19

TiDB Ansible version: 2.1.19

14.10.1.1 TiDB

- SQL Optimizer
 - Optimize the scenario of `select max(_tidb_rowid) from t` to avoid full table scan [#13294](#)
 - Fix the incorrect results caused by the incorrect value assigned to the user variable in the query and the push-down of predicates [#13230](#)
 - Fix the issue that the statistics are not accurate because a data race occurs when statistics are updated [#13690](#)
 - Fix the issue that the result is incorrect when the `UPDATE` statement contains both a sub-query and a stored generated column; fix the `UPDATE` statement execution error when this statement contains two same-named tables from different databases [#13357](#)
 - Fix the issue that the query plan might be incorrectly selected because the `PhysicalUnionScan` operator incorrectly sets the statistics [#14134](#)
 - Remove the `minAutoAnalyzeRatio` constraint to make the automatic `ANALYZE` more timely [#14013](#)
 - Fix the issue that the estimated number of rows is greater than 1 when the `WHERE` clause contains an equal condition on the unique key [#13385](#)
- SQL Execution Engine
 - Fix the precision overflow when using `int64` as the intermediate result of `unit64` in `ConvertJSONToInt` [#13036](#)
 - Fix the issue that when the `SLEEP` function is in a query (for example, `select → 1 from (select sleep(1))t;`), column pruning causes `sleep(1)` in the query to be invalid [#13039](#)
 - Reduce memory overhead by reusing `Chunk` in the `INSERT ON DUPLICATE UPDATE` statement [#12999](#)
 - Add more transaction-related fields for the `slow_query` table [#13129](#):

- * Prewrite_time
- * Commit_time
- * Get_commit_ts_time
- * Commit_backoff_time
- * Backoff_types
- * Resolve_lock_time
- * Local_latch_wait_time
- * Write_key
- * Write_size
- * Prewrite_region
- * Txn_retry

- Fix the issue that a subquery contained in an UPDATE statement is incorrectly converted; fix the UPDATE execution failure when the WHERE clause contains a subquery [#13120](#)
- Support executing ADMIN CHECK TABLE on partitioned tables [#13143](#)
- Fix the issue that the precision of statements such as SHOW CREATE TABLE is incomplete when ON UPDATE CURRENT_TIMESTAMP is used as a column attribute and floating point precision is specified [#12462](#)
- Fix the panic occurred when executing the SELECT * FROM information_schema → .KEY_COLUMN_USAGE statement because the foreign key is not checked when dropping, modifying or changing the column [#14162](#)
- Fix the issue that the returned data might be duplicated when Streaming is enabled in TiDB [#13255](#)
- Fix the Invalid time format error caused by daylight saving time [#13624](#)
- Fix the issue that data is incorrect because the precision is lost when an integer is converted to an unsigned floating point or decimal type [#13756](#)
- Fix the issue that an incorrect type of value is returned when the Quote function handles the NULL value [#13681](#)
- Fix the issue that the timezone is incorrect after parsing the date from strings using gotime.Local [#13792](#)
- Fix the issue that the result might be incorrect because the binSearch function does not return an error in the implementation of builtinIntervalRealSig [#13768](#)
- Fix the issue that an error might occur when converting the string type to the floating point type in the INSERT statement execution [#14009](#)
- Fix the incorrect result returned from the sum(distinct) function [#13041](#)
- Fix the issue that data too long is returned when CAST converting the data in union of the same location to the merged type because the returned type length of the jsonUnquoteFunction function is given an incorrect value [#13645](#)
- Fix the issue that the password cannot be set because the privilege check is too strict [#13805](#)
- Server
 - Fix the issue that KILL CONNECTION might cause the goroutine leak [#13252](#)

- Support getting the binlog status of all TiDB nodes via the `info/all` interface of the HTTP API [#13188](#)
- Fix the failure to build the TiDB project on Windows [#13650](#)
- Add the `server-version` configuration item to control and modify the version of TiDB server [#13904](#)
- Fix the issue that the binary plugin compiled with Go1.13 does not run normally [#13527](#)

- DDL

- Use the table's `COLLATE` instead of the system's default charset in the column when a table is created and the table contains `COLLATE` [#13190](#)
- Limit the length of the index name when creating a table [#13311](#)
- Fix the issue that the length of the table name is not checked when renaming a table [#13345](#)
- Check the width range of the `BIT` column [#13511](#)
- Make the error information output from `change/modify column` more understandable [#13798](#)
- Fix the issue that when executing the `drop column` operation that has not yet been handled by the downstream Drainer, the downstream might receive DML operations without the affected column [#13974](#)

14.10.1.2 TiKV

- Raftstore

- Fix the panic occurred when restarting TiKV and `is_merging` is given an incorrect value in the process of merging Regions and applying the Compact log [#5884](#)

- Importer

- Remove the limit on the gRPC message length [#5809](#)

14.10.1.3 PD

- Improve the performance of the HTTP API for getting all Regions [#1988](#)
- Upgrade etcd to fix the issue that etcd PreVote cannot elect a leader (downgrade not supported) [#2052](#)

14.10.1.4 Tools

- TiDB Binlog

- Optimize the node status information output through binlogctl [#777](#)

- Fix the panic occurred because of the `nil` value in the Drainer filter configuration [#802](#)
- Optimize the `Graceful` exit of Pump [#825](#)
- Add more detailed monitoring metrics when Pump writes binlog data [#830](#)
- Optimize Drainer's logic to refresh table information after Drainer has executed a DDL operation [#836](#)
- Fix the issue that the commit binlog of a DDL operation is ignored when Pump does not receive this binlog [#855](#)

14.10.1.5 TiDB Ansible

- Rename the `Uncommon Error` OPM monitoring item of TiDB service to `Write Binlog ↳ Error` and add the corresponding alert message [#1038](#)
- Upgrade TiSpark to 2.1.8 [#1063](#)

14.10.2 TiDB 2.1.18 Release Notes

Release date: November 4, 2019

TiDB version: 2.1.18

TiDB Ansible version: 2.1.18

14.10.2.1 TiDB

- SQL Optimizer
 - Fix the issue that invalid query ranges might appear when split by feedback [#12172](#)
 - Fix the issue that the privilege check is incorrect in point get plan [#12341](#)
 - Optimize execution performance of the `select ... limit ... offset ...` statement by pushing the Limit operator down to the `IndexLookUpReader` execution logic [#12380](#)
 - Support using parameters in `ORDER BY`, `GROUP BY` and `LIMIT OFFSET` [#12514](#)
 - Fix the issue that `IndexJoin` on the partition table returns incorrect results [#12713](#)
 - Fix the issue that the `str_to_date` function in TiDB returns a different result from MySQL when the date string and the format string do not match [#12757](#)
 - Fix the issue that outer join is incorrectly converted to inner join when the `cast` function is included in the query conditions [#12791](#)
 - Fix incorrect expression passing in the join condition of `AntiSemiJoin` [#12800](#)
- SQL Engine
 - Fix the incorrectly rounding of time (for example, `2019-09-11 11:17:47.999999666` → should be rounded to `2019-09-11 11:17:48`) [#12259](#)

- Fix the issue that the duration by `sql_type` for the PREPARE statement is not shown in the monitoring record [#12329](#)
- Fix the panic issue when the `from_unixtime` function handles null [#12572](#)
- Fix the compatibility issue that when an invalid value is inserted as the YEAR type, the result is NULL instead of 0000 [#12744](#)
- Improve the behavior of the `AutoIncrement` column when it is implicitly allocated, to keep it consistent with the default mode of MySQL auto-increment locking (“consecutive” lock mode): for the implicit allocation of multiple `AutoIncrement` IDs in a single-line `Insert` statement, TiDB guarantees the continuity of the allocated values. This improvement ensures that the JDBC `getGeneratedKeys()` method will get the correct results in any scenario [#12619](#)
- Fix the issue that the query is hanged when `HashAgg` serves as a child node of `Apply` [#12769](#)
- Fix the issue that the AND and OR logical expressions return incorrect results when it comes to type conversion [#12813](#)
- Server
 - Fix the issue that the `SLEEP()` function is invalid for the `KILL TIDB QUERY` statements [#12159](#)
 - Fix the issue that no error is reported when `AUTO_INCREMENT` incorrectly allocates `MAX int64` and `MAX uint64` [#12210](#)
 - Fix the issue that the slow query logs are not recorded when the log level is `ERROR` [#12373](#)
 - Adjust the number of times that TiDB caches schema changes and corresponding changed table information from 100 to 1024, and support modification by using the `tidb_max_delta_schema_count` system variable [#12515](#)
 - Change the query start time from the point of “starting to execute” to “starting to compile” to make SQL statistics more accurate [#12638](#)
 - Add the record of `set session autocommit` in TiDB logs [#12568](#)
 - Record SQL query start time in `SessionVars` to prevent it from being reset during plan execution [#12676](#)
 - Support ? placeholder in `ORDER BY`, `GROUP BY` and `LIMIT OFFSET` [#12514](#)
 - Add the `Prev_stmt` field in slow query logs to output the previous statement when the last statement is `COMMIT` [#12724](#)
 - Record the last statement before `COMMIT` into the log when the `COMMIT` fails in an explicitly committed transaction [#12747](#)
 - Optimize the saving method of the previous statement when the TiDB server executes a SQL statement to improve performance [#12751](#)
 - Fix the panic issue caused by `FLUSH PRIVILEGES` statements under the `skip→ grant-table=true` configuration [#12816](#)
 - Increase the default minimum step of applying AutoID from 1000 to 30000 to avoid performance bottleneck when there are many write requests in a short time [#12891](#)
 - Fix the issue that the failed `Prepared` statement is not print in the error log when TiDB panics [#12954](#)

- Fix the issue that the `COM_STMT_FETCH` time record in slow query logs is inconsistent with that in MySQL [#12953](#)
- Add an error code in the error message for write conflicts to quickly locate the cause [#12878](#)
- DDL
 - Disallow dropping the `AUTO_INCREMENT` attribute of a column by default. Modify the value of the `tidb_allow_remove_auto_inc` variable if you do need to drop this attribute. See [System Variables](#) for more details. [#12146](#)
 - Support multiple `uniques` when creating a unique index in the `Create Table` statement [#12469](#)
 - Fix a compatibility issue that if the foreign key constraint in `CREATE TABLE` statement has no schema, schema of the created table should be used instead of returning a `No Database selected` error [#12678](#)
 - Fix the issue that the `invalid list index` error is reported when executing `ADMIN CANCEL JOBS` [#12681](#)
- Monitor
 - Add types for backoff monitoring and supplement the backoff time that is not recorded before, such as the backoff time when committing [#12326](#)
 - Add a new metric to monitor `Add Index` operation progress [#12389](#)

14.10.2.2 PD

- Improve the `--help` command output of `pd-ctl` [#1772](#)

14.10.2.3 Tools

- TiDB Binlog
 - Fix the issue that `ALTER DATABASE` related DDL operations cause Drainer to exit abnormally [#770](#)
 - Support querying the transaction status information for Commit binlog to improve replication efficiency [#761](#)
 - Fix the issue that a Pump panic might occur when Drainer's `start_ts` is greater than Pump's largest `commit_ts` [#759](#)

14.10.2.4 TiDB Ansible

- Add two monitoring items “queue size” and “query histogram” for TiDB Binlog [#952](#)
- Update TiDB alerting rules [#961](#)
- Check the configuration file before the deployment and upgrade [#973](#)
- Add a new metric to monitor index speed in TiDB [#987](#)
- Update TiDB Binlog monitoring dashboard to make it compatible with Grafana v4.6.3 [#993](#)

14.10.3 TiDB 2.1.17 Release Notes

Release date: September 11, 2019

TiDB version: 2.1.17

TiDB Ansible version: 2.1.17

- New features
 - Add the `WHERE` clause in TiDB's `SHOW TABLE REGIONS` syntax
 - Add the `config-check` feature in TiKV and PD to check the configuration items
 - Add the `remove-tombstone` command in `pd-ctl` to clear tombstone store records
 - Add the `worker-count` and `txn-batch` configuration items in Reparo to control the recovery speed
- Improvements
 - Optimize PD's scheduling process by supporting actively pushing operators
 - Optimize TiKV's starting process to reduce jitters caused by restarting nodes
- Changed behaviors
 - Change `start_ts` in TiDB slow query logs from the last retry time to the first execution time
 - Replace the `Index_ids` field in TiDB slow query logs with the `Index_names` field to improve the usability of slow query logs
 - Add the `split-region-max-num` parameter in TiDB's configuration files to modify the maximum number of Regions allowed by the `SPLIT TABLE` syntax, which is increased from 1,000 to 10,000 in the default configuration

14.10.3.1 TiDB

- SQL Optimizer
 - Fix the issue that the error message is not returned correctly when an error occurs during `EvalSubquery` building `Executor` [#11811](#)
 - Fix the issue that the query result might be incorrect when the number of rows in the outer table is greater than that in a single batch in Index Lookup Join; expand the functional scope of Index Lookup Join; `UnionScan` can be used as a subnode of `IndexJoin` [#11843](#)
 - Add the display of invalid keys (like `invalid encoded key flag 252`) in the `SHOW STAT_BUCKETS` syntax, for the situation where invalid keys might occur during the statistics feedback process [#12098](#)
- SQL Execution Engine
 - Fix some incorrect results (like `select cast(13835058000000000000 as → double)`) caused by the number value that is first converted to `UINT` when the `CAST` function is converting the number value type [#11712](#)

- Fix the issue that the calculation result might be incorrect when the dividend of the DIV calculation is a decimal and this calculation contains a negative number [#11812](#)
- Add the `ConvertStrToIntStrict` function to fix the MySQL incompatibility issue caused by some strings being converted to the INT type when executing the `SELECT/EXPLAIN` statement [#11892](#)
- Fix the issue that the `Explain` result might be incorrect caused by wrong configuration of `stmtCtx` when `EXPLAIN ... FOR CONNECTION` is used [#11978](#)
- Fix the issue that the result returned by the `unaryMinus` function is incompatible with MySQL, caused by the non-decimal result when the integer result overflows [#11990](#)
- Fix the issue that `last_insert_id()` might be incorrect, caused by the counting order when the `LOAD DATA` statement is being executed [#11994](#)
- Fix the issue that `last_insert_id()` might be incorrect when the user writes auto-increment column data in an explicit-implicit mixed way [#12001](#)
- Fix an over-quoted bug for the `JSON_UNQUOTE` function: only values enclosed by double quote marks ("") should be unquoted. For example, the result of “`SELECT → JSON_UNQUOTE("\\\\")`” should be “\\” (not changed) [#12096](#)
- Server
 - Change `start_ts` recorded in slow query logs from the last retry time to the first execution time when retrying TiDB transactions [#11878](#)
 - Add the number of keys of a transaction in `LockResolver` to avoid the scan operation on the whole Region and reduce costs of resolving locking when the number of keys is reduced [#11889](#)
 - Fix the issue that the `succ` field value might be incorrect in slow query logs [#11886](#)
 - Replace the `Index_ids` filed in slow query logs with the `Index_names` field to improve the usability of slow query logs [#12063](#)
 - Fix the connection break issue caused by TiDB parsing - into EOF Error when `Duration` contains - (like `select time('--')`) [#11910](#)
 - Remove an invalid Region from `RegionCache` more quickly to reduce the number of requests sent to this Region [#11931](#)
 - Fix the connection break issue caused by incorrectly handling the OOM panic issue when `oom-action = "cancel"` and OOM occurs in the `Insert Into ... → Select` syntax [#12126](#)
- DDL
 - Add the reverse scan interface for `tikvSnapshot` to efficiently query DDL history jobs. After using this interface, the execution time of `ADMIN SHOW DDL JOBS` is remarkably decreased [#11789](#)
 - Improve the `CREATE TABLE ... PRE_SPLIT_REGION` syntax: change the number of pre-splitting Regions from 2^{N-1} to 2^N when `PRE_SPLIT_REGION = N` [#11797](#)

- Decrease the default parameter value for the background worker thread of the `Add Index` operation to avoid great impacts on online workloads [#11875](#)
- Improve the `SPLIT TABLE` syntax behavior: generate N data Region(s) and one index Region when `SPLIT TABLE ... REGIONS N` is used to divide Regions [#11929](#)
- Add the `split-region-max-num` parameter (10000 by default) in the configuration file to make the maximum number of Regions allowed by the `SPLIT TABLE` syntax adjustable [#12080](#)
- Fix the issue that the `CREATE TABLE` clause cannot be parsed by the downstream MySQL, caused by uncommented `PRE_SPLIT_REGIONS` in this clause when the system writes the binlog [#12121](#)
- Add the `WHERE` sub-clause in `SHOW TABLE ... REGIONS` and `SHOW TABLE ... INDEX ... REGIONS` [#12124](#)
- Monitor
 - Add the `connection_transient_failure_count` monitoring metric to count gRPC connection errors of `tikvclient` [#12092](#)

14.10.3.2 TiKV

- Fix the incorrect result of counting keys in a Region in some cases [#5415](#)
- Add the `config-check` option in TiKV to check whether the TiKV configuration item is valid [#5391](#)
- Optimize the starting process to reduce jitters caused by restarting nodes [#5277](#)
- Optimize the resolving locking process in some cases to speed up resolving locking for transactions [#5339](#)
- Optimize the `get_txn_commit_info` process to speed up committing transactions [#5062](#)
- Simplify Raft-related logs [#5425](#)
- Resolve the issue that TiKV exits abnormally in some cases [#5441](#)

14.10.3.3 PD

- Add the `config-check` option in PD to check whether the PD configuration item is valid [#1725](#)
- Add the `remove-tombstone` command in pd-ctl to support clearing tombstone store records [#1705](#)
- Support actively pushing operators to speed up scheduling [#1686](#)

14.10.3.4 Tools

- TiDB Binlog
 - Add `worker-count` and `txn-batch` configuration items in Reparo to control the recovery speed [#746](#)

- Optimize the memory usage of Drainer to improve the parallel execution efficiency [#735](#)
- Fix the bug that Pump cannot quit normally in some cases [#739](#)
- Optimize the processing logic of LevelDB in Pump to improve the execution efficiency of GC [#720](#)
- TiDB Lightning
 - Fix the bug that tidb-lightning might crash caused by re-importing data from the checkpoint [#239](#)

14.10.3.5 TiDB Ansible

- Update the Spark version to 2.4.3, and update the TiSpark version to 2.2.0 that is compatible with Spark 2.4.3 [#914](#), [#919](#)
- Fix the issue that there is a long waiting time when the remote machine password has expired [#937](#), [#948](#)

14.10.4 TiDB 2.1.16 Release Notes

Release date: August 15, 2019

TiDB version: 2.1.16

TiDB Ansible version: 2.1.16

14.10.4.1 TiDB

- SQL Optimizer
 - Fix the issue that row count is estimated inaccurately for the equal condition on the time column [#11526](#)
 - Fix the issue that TIDB_INLJ Hint does not take effect or take effect on the specified table [#11361](#)
 - Change the implementation of NOT EXISTS in a query from OUTER JOIN to ANTI JOIN to find a more optimized execution plan [#11291](#)
 - Support subqueries within SHOW statements, allowing syntaxes such as SHOW ↪ COLUMNS FROM tbl WHERE FIELDS IN (SELECT 'a') [#11461](#)
 - Fix the issue that the SELECT ... CASE WHEN ... ELSE NULL ... query gets an incorrect result caused by the constant folding optimization [#11441](#)
- SQL Execution Engine
 - Fix the issue that the DATE_ADD function gets a wrong result when INTERVAL is negative [#11616](#)
 - Fix the issue that the DATE_ADD function might get an incorrect result because it performs type conversion wrongly when it accepts an argument of the FLOAT, DOUBLE, or DECIMAL type [#11628](#)

- Fix the issue that the error message is inaccurate when CAST(JSON AS SIGNED) overflows [#11562](#)
- Fix the issue that other child nodes are not closed when one child node fails to be closed and returns an error during the process of closing Executor [#11598](#)
- Support SPLIT TABLE statements that return the number of Regions that are successfully split and a finished percentage rather than an error when the scheduling is not finished for Region scatter before the timeout [#11487](#)
- Make REGEXP BINARY function case sensitive to be compatible with MySQL [#11505](#)
- Fix the issue that NULL is not returned correctly because the value of YEAR in the DATE_ADD/DATE_SUB result overflows when it is smaller than 0 or larger than 65535 [#11477](#)
- Add in the slow query table a Succ field that indicates whether the execution succeeds [#11412](#)
- Fix the MySQL incompatibility issue caused by fetching the current timestamp multiple times when a SQL statement involves calculations of the current time (such as CURRENT_TIMESTAMP or NOW) [#11392](#)
- Fix the issue that the AUTO_INCREMENT columns do not handle the FLOAT or DOUBLE type [#11389](#)
- Fix the issue that NULL is not returned correctly when the CONVERT_TZ function accepts an invalid argument [#11357](#)
- Fix the issue that an error is reported by the PARTITION BY LIST statement. (Currently only the syntax is supported; when TiDB executes the statement, a regular table is created and a prompting message is provided) [#11236](#)
- Fix the issue that Mod(%), Multiple(*), and Minus(-) operations return an inconsistent 0 result with that in MySQL when there are many decimal digits (such as select 0.000 % 0.11234500000000000000) [#11353](#)
- Server
 - Fix the issue that the plugin gets a NULL domain when OnInit is called back [#11426](#)
 - Fix the issue that the table information in a schema can still be obtained through the HTTP interface after the schema has been deleted [#11586](#)
- DDL
 - Disallow dropping indexes on auto-increment columns to avoid incorrect results of the auto-increment columns caused by this operation [#11402](#)
 - Fix the issue that the character set of the column is not correct when creating and modifying the table with different character sets and collations [#11423](#)
 - Fix the issue that the column schema might get wrong when alter table ... ↳ set default... and another DDL statement that modifies this column are executed in parallel [#11374](#)
 - Fix the issue that data fails to be backfilled when Generated Column A depends on Generated Column B and A is used to create an index [#11538](#)
 - Speed up ADMIN CHECK TABLE operations [#11538](#)

14.10.4.2 TiKV

- Support returning an error message when the client accesses a TiKV Region that is being closed [#4820](#)
- Support reverse `raw_scan` and `raw_batch_scan` interfaces [#5148](#)

14.10.4.3 Tools

- TiDB Binlog
 - Add the `ignore-txn-commit-ts` configuration item in Drainer to skip executing some statements in a transaction [#697](#)
 - Add the configuration item check on startup, which stops Pump and Drainer from running and returns an error message when meeting invalid configuration items [#708](#)
 - Add the `node-id` configuration in Drainer to specify Drainer's node ID [#706](#)
- TiDB Lightning
 - Fix the issue that `tikv_gc_life_time` fails to be changed back to its original value when 2 checksums are running at the same time [#224](#)

14.10.4.4 TiDB Ansible

- Add the `log4j` configuration file in Spark [#842](#)
- Update the tispark jar package to v2.1.2 [#863](#)
- Fix the issue that the Prometheus configuration file is generated in the wrong format when TiDB Binlog uses Kafka or ZooKeeper [#845](#)
- Fix the bug that PD fails to switch the Leader when executing the `rolling_update.` → `yml` operation [#888](#)
- Optimize the logic of rolling updating PD nodes - upgrade Followers first and then the Leader - to improve stability [#895](#)

14.10.5 TiDB 2.1.15 Release Notes

Release date: July 16, 2019

TiDB version: 2.1.15

TiDB Ansible version: 2.1.15

14.10.5.1 TiDB

- Fix the issue that the DATE_ADD function returns wrong results due to incorrect alignment when dealing with microseconds [#11289](#)
- Fix the issue that an error is reported when the empty value in the string column is compared with FLOAT or INT [#11279](#)
- Fix the issue that the INSERT function fails to correctly return the NULL value when a parameter is NULL [#11249](#)
- Fix the issue that an error occurs when indexing the column of the non-string type and 0 length [#11215](#)
- Add the SHOW TABLE REGIONS statement to query the Region distribution of a table through SQL statements [#11238](#)
- Fix the issue that an error is reported when using the UPDATE ... SELECT statement because the projection elimination is used to optimize rules in the SELECT subqueries [#11254](#)
- Add the ADMIN PLUGINS ENABLE/ADMIN PLUGINS DISABLE SQL statement to dynamically enable or disable plugins [#11189](#)
- Add the session connection information in the Audit plugin [#11189](#)
- Fix the panic issue that happens when a column is queried on multiple times and the returned result is NULL during point queries [#11227](#)
- Add the tidb_scatter_region configuration item to scatter table Regions when creating a table [#11213](#)
- Fix the data race issue caused by non-thread safe rand.Rand when using the RAND function [#11170](#)
- Fix the issue that the comparison result of integers and non-integers is incorrect in some cases [#11191](#)
- Support modifying the collation of a database or a table, but the character set of the database/table has to be UTF-8 or utf8mb4 [#11085](#)
- Fix the issue that the precision shown by the SHOW CREATE TABLE statement is incomplete when CURRENT_TIMESTAMP is used as the default value of the column and the float precision is specified [#11087](#)

14.10.5.2 TiKV

- Unify the log format [#5083](#)
- Improve the accuracy of Region's approximate size or keys in extreme cases to improve the accuracy of scheduling [#5085](#)

14.10.5.3 PD

- Unify the log format [#1625](#)

14.10.5.4 Tools

TiDB Binlog

- Optimize the Pump GC strategy and remove the restriction that the unconsumed binlog cannot be cleaned to make sure that the resources are not occupied for a long time [#663](#)

TiDB Lightning

- Fix the import error that happens when the column names specified by the SQL dump are not in lowercase [#210](#)

14.10.5.5 TiDB Ansible

- Add the `parse duration` and `compile duration` monitoring items in TiDB Dashboard to monitor the time that it takes to parse SQL statements and execute compilation [#815](#)

14.10.6 TiDB 2.1.14 Release Notes

Release date: July 04, 2019

TiDB version: 2.1.14

TiDB Ansible version: 2.1.14

14.10.6.1 TiDB

- Fix wrong query results caused by column pruning in some cases [#11019](#)
- Fix the wrongly displayed information in `db` and `info` columns of `show processlist` [#11000](#)
- Fix the issue that `MAX_EXECUTION_TIME` as a SQL hint and global variable does not work in some cases [#10999](#)
- Support automatically adjust the incremental gap allocated by auto-increment ID based on the load [#10997](#)
- Fix the issue that the `Distsql` memory information of `MemTracker` is not correctly cleaned when a query ends [#10971](#)
- Add the `MEM` column in the `information_schema.processlist` table to describe the memory usage of a query [#10896](#)
- Add the `max_execution_time` global system variable to control the maximum execution time of a query [#10940](#)
- Fix the panic caused by using unsupported aggregate functions [#10911](#)
- Add an automatic rollback feature for the last transaction when the `load data` statement fails [#10862](#)

- Fix the issue that TiDB returns a wrong result in some cases when the `OOMAction` configuration item is set to `Cancel` [#11016](#)
- Disable the `TRACE` statement to avoid the TiDB panic issue [#11039](#)
- Add the `mysql.expr_pushdown_blacklist` system table that dynamically enables/disables pushing down specific functions to Coprocessor [#10998](#)
- Fix the issue that the `ANY_VALUE` function does not work in the `ONLY_FULL_GROUP_BY` mode [#10994](#)
- Fix the incorrect evaluation caused by not doing a deep copy when evaluating the user variable of the string type [#11043](#)

14.10.6.2 TiKV

- Optimize processing the empty callback when processing the Raftstore message to avoid sending unnecessary message [#4682](#)

14.10.6.3 PD

- Adjust the log output level from `Error` to `Warning` when reading an invalid configuration item [#1577](#)

14.10.6.4 Tools

TiDB Binlog

- Reparo
 - Add the `safe-mode` configuration item, and support importing duplicated data after this item is enabled [#662](#)
- Pump
 - Add the `stop-write-at-available-space` configuration item to limit the available binlog space [#659](#)
 - Fix the issue that Garbage Collector does not work sometimes when the number of LevelDB L0 files is 0 [#648](#)
 - Optimize the algorithm of deleting log files to speed up releasing the space [#648](#)
- Drainer
 - Fix the failure to update BIT columns in the downstream [#655](#)

14.10.6.5 TiDB Ansible

- Add the precheck feature for the `ansible` command and its `jmespath` and `jinja2` dependency packages [#807](#)
- Add the `stop-write-at-available-space` parameter (10 GiB by default) in Pump, and stop writing binlog files in Pump when the available disk space is less than the parameter value [#807](#)

14.10.7 TiDB 2.1.13 Release Notes

Release date: June 21, 2019

TiDB version: 2.1.13

TiDB Ansible version: 2.1.13

14.10.7.1 TiDB

- Add a feature to use SHARD_ROW_ID_BITS to scatter row IDs when the column contains an AUTO_INCREMENT attribute to relieve the hotspot issue [#10788](#)
- Optimize the lifetime of invalid DDL metadata to speed up recovering the normal execution of DDL operations after upgrading the TiDB cluster [#10789](#)
- Fix the OOM issue in high concurrent scenarios caused by the failure to quickly release Coprocessor resources, resulted from the `execdetails.ExecDetails` pointer [#10833](#)
- Add the `update-stats` configuration item to control whether to update statistics [#10772](#)
- Add the following TiDB-specific syntax to support Region presplit to solve the hotspot issue:
 - Add the `PRE_SPLIT_REGIONS` table option [#10863](#)
 - Add the `SPLIT TABLE table_name INDEX index_name` syntax [#10865](#)
 - Add the `SPLIT TABLE [table_name] BETWEEN (min_value...) AND (max_value → ...)REGIONS [region_num]` syntax [#10882](#)
- Fix the panic issue caused by the `KILL` syntax in some cases [#10879](#)
- Improve the compatibility with MySQL for `ADD_DATE` in some cases [#10718](#)
- Fix the wrong estimation for the selectivity rate of the inner table selection in index join [#10856](#)

14.10.7.2 TiKV

- Fix the issue that incomplete snapshots are generated in the system caused by the iterator not checking the status [#4940](#)
- Add a feature to check the validity for the `block-size` configuration [#4930](#)

14.10.7.3 Tools

- TiDB Binlog
 - Fix the wrong offset issue caused by Pump not checking the returned value when it fails to write data [#640](#)
 - Add the `advertise-addr` configuration in Drainer to support the bridge mode in the container environment [#634](#)

14.10.8 TiDB 2.1.12 Release Notes

Release date: June 13, 2019

TiDB version: 2.1.12

TiDB Ansible version: 2.1.12

14.10.8.1 TiDB

- Fix the issue caused by unmatched data types when using the index query feedback [#10755](#)
- Fix the issue that the blob column is changed to the text column caused by charset altering in some cases [#10745](#)
- Fix the issue that the GRANT operation in the transaction mistakenly reports “Duplicate Entry” in some cases [#10739](#)
- Improve the compatibility with MySQL of the following features
 - The DAYNAME function [#10732](#)
 - The MONTHNAME function [#10733](#)
 - Support the 0 value for the EXTRACT function when processing MONTH [#10702](#)
 - The DECIMAL type can be converted to TIMESTAMP or DATETIME [#10734](#)
- Change the column charset while changing the table charset [#10714](#)
- Fix the overflow issue when converting a decimal to a float in some cases [#10730](#)
- Fix the issue that some extremely large messages report the “grpc: received message larger than max” error caused by inconsistent maximum sizes of messages sent/received by gRPC of TiDB and TiKV [#10710](#)
- Fix the panic issue caused by ORDER BY not filtering NULL in some cases [#10488](#)
- Fix the issue that values returned by the UUID function might be duplicate when multiple nodes exist [#10711](#)
- Change the value returned by CAST(-num as datetime) from error to NULL [#10703](#)
- Fix the issue that an unsigned histogram meets signed ranges in some cases [#10695](#)
- Fix the issue that an error is reported mistakenly for reading data when the statistics feedback meets the bigint unsigned primary key [#10307](#)
- Fix the issue that the result of Show Create Table for partitioned tables is not correctly displayed in some cases [#10690](#)
- Fix the issue that the calculation result of the GROUP_CONCAT aggregate function is not correct for some correlated subqueries [#10670](#)
- Fix the issue that the result is wrongly displayed when the memory table of slow queries parses the slow query log in some cases [#10776](#)

14.10.8.2 PD

- Fix the issue that etcd leader election is blocked in extreme conditions [#1576](#)

14.10.8.3 TiKV

- Fix the issue that Regions are not available during the leader transfer process in extreme conditions [#4799](#)
- Fix the issue that TiKV loses data when the power of the machine fails abnormally, caused by delayed data flush to the disk when receiving snapshots [#4850](#)

14.10.9 TiDB 2.1.11 Release Notes

Release date: June 03, 2019

TiDB version: 2.1.11

TiDB Ansible version: 2.1.11

14.10.9.1 TiDB

- Fix the issue that incorrect schema is used for `delete from join` [#10595](#)
- Fix the issue that the built-in `CONVERT()` may return incorrect field type [#10263](#)
- Merge non-overlapped feedback when updating bucket count [#10569](#)
- Fix calculation errors of `unix_timestamp() - unix_timestamp(now())` [#10491](#)
- Fix the incompatibility issue of `period_diff` with MySQL 8.0 [#10501](#)
- Skip `Virtual Column` when collecting statistics to avoid exceptions [#10628](#)
- Support the `SHOW OPEN TABLES` statement [#10374](#)
- Fix the issue that goroutine leak may happen in some cases [#10656](#)
- Fix the issue that setting the `tidb_snapshot` variable in some cases may cause incorrect parsing of time format [#10637](#)

14.10.9.2 PD

- Fix the issue that hot Region may fail to be scheduled due to `balance-region` [#1551](#)
- Set hotspot related scheduling priorities to high [#1551](#)
- Add two configuration items [#1551](#)
 - `hot-region-schedule-limit` to control the maximum number of concurrent hotspot scheduling tasks
 - `hot-region-cache-hits-threshold` to identify a hot Region

14.10.9.3 TiKV

- Fix the issue that the learner reads an empty index when there is only one leader and one learner [#4751](#)
- Process `ScanLock` and `ResolveLock` in the thread pool with a high priority to reduce their impacts on commands with a normal priority [#4791](#)
- Sync all files of received snapshots [#4811](#)

14.10.9.4 Tools

- TiDB Binlog
 - Limit data deletion speed during GC to avoid QPS degrading caused by [WritePause #620](#)

14.10.9.5 TiDB Ansible

- Add Drainer parameters [#760](#)

14.10.10 TiDB 2.1.10 Release Notes

Release date: May 22, 2019

TiDB version: 2.1.10

TiDB Ansible version: 2.1.10

14.10.10.1 TiDB

- Fix the issue that some abnormalities cause incorrect table schema when using `tidb_snapshot` to read the history data [#10359](#)
- Fix the issue that the `NOT` function causes wrong read results in some cases [#10363](#)
- Fix the wrong behavior of `Generated Column` in the `Replace` or `Insert on → duplicate update` statement [#10385](#)
- Fix a bug of the `BETWEEN` function in the `DATE/DATETIME` comparison [#10407](#)
- Fix the issue that a single line of a slow log that is too long causes an error report when using the `SLOW_QUERY` table to query a slow log [#10412](#)
- Fix the issue that the result of `DATETIME plus INTERVAL` is not the same with that of MySQL in some cases [#10416, #10418](#)
- Add the check for the invalid time of February in a leap year [#10417](#)
- Execute the internal initialization operation limitation only in the DDL owner to avoid a large number of conflict error reports when initializing the cluster [#10426](#)
- Fix the issue that `DESC` is incompatible with MySQL when the default value of the output timestamp column is `default current_timestamp on update → current_timestamp` [#10337](#)
- Fix the issue that an error occurs during the privilege check in the `Update` statement [#10439](#)
- Fix the issue that wrong calculation of `RANGE` causes a wrong result in the `CHAR` column in some cases [#10455](#)
- Fix the issue that the data might be overwritten after decreasing `SHARD_ROW_ID_BITS` [#9868](#)
- Fix the issue that `ORDER BY RAND()` does not return random numbers [#10064](#)
- Prohibit the `ALTER` statement modifying the precision of decimals [#10458](#)

- Fix the compatibility issue of the `TIME_FORMAT` function with MySQL [#10474](#)
- Check the parameter validity of `PERIOD_ADD` [#10430](#)
- Fix the issue that the behavior of the invalid `YEAR` string in TiDB is incompatible with that in MySQL [#10493](#)
- Support the `ALTER DATABASE` syntax [#10503](#)
- Fix the issue that the `SLOW_QUERY` memory engine reports an error when no ; exists in the slow query statement [#10536](#)
- Fix the issue that the `Add index` operation in partitioned tables cannot be canceled in some cases [#10533](#)
- Fix the issue that the OOM panic cannot be recovered in some cases [#10545](#)
- Improve the security of the DDL operation rewriting the table metadata [#10547](#)

14.10.10.2 PD

- Fix the issue that the priority of the leader does not take effect [#1533](#)

14.10.10.3 TiKV

- Reject transferring the leader in a Region whose configuration has been changed recently to avoid transfer failure [#4684](#)
- Add the priority label for Coprocessor metrics [#4643](#)
- Fix the possible dirty read issue during transferring the leader [#4724](#)
- Fix the issue that `CommitMerge` causes the restart failure of TiKV in some cases [#4615](#)
- Fix unknown logs [#4730](#)

14.10.10.4 Tools

- TiDB Lightning
 - Add the retry feature when TiDB Lightning fails to send data to `importer` [#176](#)
- TiDB Binlog
 - Optimize the Pump storage log to facilitate troubleshooting [#607](#)

14.10.10.5 TiDB Ansible

- Update the configuration file of TiDB Lightning and add the `tidb_lightning_ctl` script [#d3a4a368](#)

14.10.11 TiDB 2.1.9 Release Notes

Release date: May 6, 2019

TiDB version: 2.1.9

TiDB Ansible version: 2.1.9

14.10.11.1 TiDB

- Fix compatibility of the `MAKETIME` function when unsigned type overflows [#10089](#)
- Fix the stack overflow caused by constant folding in some cases [#10189](#)
- Fix the privilege check issue for `Update` when an alias exists in some cases [#10157](#), [#10326](#)
- Track and control memory usage in DistSQL [#10197](#)
- Support specifying collation as `utf8mb4_0900_ai_ci` [#10201](#)
- Fix the wrong result issue of the `MAX` function when the primary key is of the Unsigned type [#10209](#)
- Fix the issue that NULL values can be inserted into NOT NULL columns in the non-strict SQL mode [#10254](#)
- Fix the wrong result issue of the `COUNT` function when multiple columns exist in `DISTINCT` [#10270](#)
- Fix the panic issue occurred when `LOAD DATA` parses irregular CSV files [#10269](#)
- Ignore the overflow error when the outer and inner join key types are inconsistent in `Index Lookup Join` [#10244](#)
- Fix the issue that a statement is wrongly judged as point-get in some cases [#10299](#)
- Fix the wrong result issue when the time type does not convert the time zone in some cases [#10345](#)
- Fix the issue that TiDB character set cases are inconsistent in some cases [#10354](#)
- Support controlling the number of rows returned by operator [#9166](#)
 - Selection & Projection [#10110](#)
 - `StreamAgg` & `HashAgg` [#10133](#)
 - `TableReader` & `IndexReader` & `IndexLookup` [#10169](#)
- Improve the slow query log
 - Add `SQL Digest` to distinguish similar SQL [#10093](#)
 - Add version information of statistics used by slow query statements [#10220](#)
 - Show memory consumption of a statement in slow query log [#10246](#)
 - Adjust the output format of Coprocessor related information so it can be parsed by `pt-query-digest` [#10300](#)
 - Fix the # character issue in slow query statements [#10275](#)
 - Add some information columns to the memory table of slow query statements [#10317](#)
 - Add the transaction commit time to slow query log [#10310](#)
 - Fix the issue some time formats cannot be parsed by `pt-query-digest` [#10323](#)

14.10.11.2 PD

- Support the `GetOperator` service [#1514](#)

14.10.11.3 TiKV

- Fix potential quorum changes when transferring leader [#4604](#)

14.10.11.4 Tools

- TiDB Binlog
 - Fix the issue that data replication is interrupted because data in the unsigned int type of primary key column are minus numbers [#574](#)
 - Remove the compression option when the downstream is pb and change the downstream name from pb to file [#597](#)
 - Fix the bug that Reparo introduced in 2.1.7 generates wrong UPDATE statements [#576](#)
- TiDB Lightning
 - Fix the bug that the bit type of column data is incorrectly parsed by the parser [#164](#)
 - Fill the lacking column data in dump files using row id or the default column value [#174](#)
 - Fix the Importer bug that some SST files fail to be imported but it still returns successful import result [#4566](#)
 - Support setting a speed limit in Importer when uploading SST files to TiKV [#4607](#)
 - Change Importer RocksDB SST compression method to lz4 to reduce CPU consumption [#4624](#)
- sync-diff-inspector
 - Support checkpoint [#227](#)

14.10.11.5 TiDB Ansible

- Update links in tidb-ansible documentation according to docs refactoring [#740](#), [#741](#)
- Remove the `enable_slow_query_log` parameter in the `inventory.ini` file and output the slow query log to a separate log file by default [#742](#)

14.10.12 TiDB 2.1.8 Release Notes

Release date: April 12, 2019

TiDB version: 2.1.8

TiDB Ansible version: 2.1.8

14.10.12.1 TiDB

- Fix the issue that the processing logic of `GROUP_CONCAT` function is incompatible with MySQL when there is a NULL-valued parameter [#9930](#)
- Fix the equality check issue of decimal values in the `Distinct` mode [#9931](#)

- Fix the collation compatibility issue of the date, datetime, and timestamp types for the SHOW FULL COLUMNS statement
 - [#9938](#)
 - [#10114](#)
- Fix the issue that the row count estimation is inaccurate when the filtering condition contains correlated columns [#9937](#)
- Fix the compatibility issue between the DATE_ADD and DATE_SUB functions
 - [#9963](#)
 - [#9966](#)
- Support the %H format for the STR_TO_DATE function to improve compatibility [#9964](#)
- Fix the issue that the result is wrong when the GROUP_CONCAT function groups by a unique index [#9969](#)
- Return a warning when the Optimizer Hints contains an unmatched table name [#9970](#)
- Unify the log format to facilitate collecting logs using tools for analysis Unified Log Format
- Fix the issue that a lot of NULL values cause inaccurate statistics estimation [#9979](#)
- Fix the issue that an error is reported when the default value of the TIMESTAMP type is the boundary value [#9987](#)
- Validate the value of time_zone [#10000](#)
- Support the 2019.01.01 time format [#10001](#)
- Fix the issue that the row count estimation is displayed incorrectly in the result returned by the EXPLAIN statement in some cases [#10044](#)
- Fix the issue that KILL TIDB [session id] cannot instantly stop the execution of a statement in some cases [#9976](#)
- Fix the predicate pushdown issue of constant filtering conditions in some cases [#10049](#)
- Fix the issue that a read-only statement is not processed correctly in some cases [#10048](#)

14.10.12.2 PD

- Fix the issue that regionScatterer might generate an invalid OperatorStep [#1482](#)
- Fix the issue that a hot store makes incorrect statistics of keys [#1487](#)
- Fix the too short timeout issue of the MergeRegion operator [#1495](#)
- Add elapsed time metrics of the PD server handling TSO requests [#1502](#)

14.10.12.3 TiKV

- Fix the issue of wrong statistics of the read traffic [#4441](#)
- Fix the raftstore performance issue when too many Regions exist [#4484](#)
- Do not ingest files when the number of level 0 SST files exceeds level_zero_slowdownWrites_trigg
→ /2 [#4464](#)

14.10.12.4 Tools

- Optimize the order of importing tables for Lightning to reduce the effects of large tables executing `Checksum` and `Analyze` on the cluster during the importing process and improve the success rate of `Checksum` and `Analyze` [#156](#)
- Improve the encoding SQL performance by 50% for Lightning by directly parsing the data source file content to `types.Datum` of TiDB to avoid additional parsing working of the KV encoder [#145](#)
- Add the `storage.sync-log` configuration item in TiDB Binlog Pump to support flushing disks of the local storage asynchronously in Pump [#529](#)
- Support traffic compression of communication between TiDB Binlog Pump and Drainer [#530](#)
- Add the `syncer.sql-mode` configuration item in TiDB Binlog Drainer to support using different `sql-modes` to parse DDL queries [#513](#)
- Add the `syncer.ignore-table` configuration item in TiDB Binlog Drainer to support filtering tables not to be replicated [#526](#)

14.10.12.5 TiDB Ansible

- Modify the version limit for the operating system and only support CentOS 7.0 or later and Red Hat 7.0 or later [#734](#)
- Add the feature of checking whether `epollexclusive` is supported in every OS [#728](#)
- Add the version limit for rolling update to prohibit upgrading a version of 2.0.1 or earlier to a version of 2.1 or later [#728](#)

14.10.13 TiDB 2.1.7 Release Notes

Release Date: March 28, 2019

TiDB version: 2.1.7

TiDB Ansible version: 2.1.7

14.10.13.1 TiDB

- Fix the issue of longer startup time when upgrading the program caused by canceling DDL operations [#9768](#)
- Fix the issue that the `check-mb4-value-in-utf8` configuration item is in the wrong position in the `config.example.toml` file [#9852](#)
- Improve the compatibility of the `str_to_date` built-in function with MySQL [#9817](#)
- Fix the compatibility issue of the `last_day` built-in function [#9750](#)
- Add the `tidb_table_id` column for `infoschema.tables` to facilitate getting `table_id` ↗ by using SQL statements and add the `tidb_indexes` system table to manage the relationship between Table and Index [#9862](#)

- Add a check about the null definition of Table Partition [#9663](#)
- Change the privileges required by Truncate Table from Delete to Drop to make it consistent with MySQL [#9876](#)
- Support using subqueries in the DO statement [#9877](#)
- Fix the issue that the default_week_format variable does not take effect in the week function [#9753](#)
- Support the plugin framework [#9880](#), [#9888](#)
- Support checking the enabling state of binlog by using the log_bin system variable [#9634](#)
- Support checking the Pump/Drainer status by using SQL statements [#9896](#)
- Fix the compatibility issue about checking mb4 character on utf8 when upgrading TiDB [#9887](#)
- Fix the panic issue when the aggregate function calculates JSON data in some cases [#9927](#)

14.10.13.2 PD

- Fix the issue that the transferring leader step cannot be created in the balance-region when the number of replicas is one [#1462](#)

14.10.13.3 Tools

- Support replicating generated columns by using binlog

14.10.13.4 TiDB Ansible

Change the default retention time of Prometheus monitoring data to 30d

14.10.14 TiDB 2.1.6 Release Notes

On March 15, 2019, TiDB 2.1.6 is released. The corresponding TiDB Ansible 2.1.6 is also released. Compared with TiDB 2.1.5, this release has greatly improved the stability, the SQL optimizer, statistics, and the execution engine.

14.10.14.1 TiDB

- SQL Optimizer/Executor
 - Optimize planner to select the outer table based on cost when both tables are specified in Hint of TIDB_INLJ [#9615](#)
 - Fix the issue that IndexScan cannot be selected correctly in some cases [#9587](#)
 - Fix incompatibility with MySQL of check in the agg function in subqueries [#9551](#)
 - Make show stats_histograms only output valid columns to avoid panics [#9502](#)

- Server
 - Support the `log_bin` variable to enable/disable Binlog [#9634](#)
 - Add a sanity check for transactions to avoid false transaction commit [#9559](#)
 - Fix the issue that setting variables may lead to panic [#9539](#)
- DDL
 - Fix the issue that the `Create Table Like` statement causes panic in some cases [#9652](#)
 - Enable the `AutoSync` feature of etcd clients to avoid connection issues between TiDB and etcd in some cases [#9600](#)

14.10.14.2 TiKV

- Fix the issue that a `protobuf` parsing failure would in some cases cause a `StoreNotMatch` error [#4303](#)

14.10.14.3 Tools

- Lightning
 - Change the default `region-split-size` of importer to 512 MiB [#4369](#)
 - Save the intermediate SST previously cached in memory to the local disk to reduce memory usage [#4369](#)
 - Limit the memory usage of RocksDB [#4369](#)
 - Fix the issue that Regions are scattered before scheduling is finished [#4369](#)
 - Separate importing of data and indexes for large tables to effectively reduce time consumption when importing in batches [#132](#)
 - Support CSV [#111](#)
 - Fix the error of import failure due to non-alphanumeric characters in schema names [#9547](#)

14.10.15 TiDB 2.1.5 Release Notes

On February 28, 2019, TiDB 2.1.5 is released. The corresponding TiDB Ansible 2.1.5 is also released. Compared with TiDB 2.1.4, this release has greatly improved the stability, the SQL optimizer, statistics, and the execution engine.

14.10.15.1 TiDB

- SQL Optimizer/Executor
 - Make `SHOW CREATE TABLE` do not print the column charset information when the charset information of a column is the same with that of a table, to improve the compatibility of `SHOW CREATE TABLE` with MySQL [#9306](#)

- Fix the panic or the wrong result of the `Sort` operator in some cases by extracting `ScalarFunc` from `Sort` to a `Projection` operator for computing to simplify the computing logic of `Sort` [#9319](#)
- Remove the sorting field with constant values in the `Sort` operator [#9335](#), [#9440](#)
- Fix the data overflow issue when inserting data into an unsigned integer column [#9339](#)
- Set `cast_as_binary` to `NULL` when the length of the target binary exceeds `max_allowed_packet` [#9349](#)
- Optimize the constant folding process of `IF` and `IFNULL` [#9351](#)
- Optimize the index selection of TiDB using skyline pruning to improve the stability of simple queries [#9356](#)
- Support computing the selectivity of the DNF expression [#9405](#)
- Fix the wrong SQL query result of `!=ANY()` and `=ALL()` in some cases [#9403](#)
- Fix the panic or the wrong result when the Join Key types of two tables on which the `Merge Join` operation is performed are different [#9438](#)
- Fix the issue that the result of the `RAND()` function is not compatible with MySQL [#9446](#)
- Refactor the logic of `Semi Join` processing `NULL` and the empty result set to get the correct result and improve the compatibility with MySQL [#9449](#)
- Server
 - Add the `tidb_constraint_check_in_place` system variable to check the data uniqueness constraint when executing the `INSERT` statement [#9401](#)
 - Fix the issue that the value of the `tidb_force_priority` system variable is different from that set in the configuration file [#9347](#)
 - Add the `current_db` field in general logs to print the name of the currently used database [#9346](#)
 - Add an HTTP API of obtaining the table information with the table ID [#9408](#)
 - Fix the issue that `LOAD DATA` loads incorrect data in some cases [#9414](#)
 - Fix the issue that it takes a long time to build a connection between the MySQL client and TiDB in some cases [#9451](#)
- DDL
 - Fix some issues when canceling the `DROP COLUMN` operation [#9352](#)
 - Fix some issues when canceling the `DROP` or `ADD` partitioned table operation [#9376](#)
 - Fix the issue that `ADMIN CHECK TABLE` mistakenly reports the data index inconsistency in some cases [#9399](#)
 - Fix the time zone issue of the `TIMESTAMP` default value [#9108](#)

14.10.15.2 PD

- Provide the `exclude_tombstone_stores` option in the `GetAllStores` interface to remove the Tombstone store from the returned result [#1444](#)

14.10.15.3 TiKV

- Fix the issue that Importer fails to import data in some cases [#4223](#)
- Fix the KeyNotInRegion error in some cases [#4125](#)
- Fix the panic issue caused by Region merge in some cases [#4235](#)
- Add the detailed StoreNotMatch error message [#3885](#)

14.10.15.4 Tools

- Lightning
 - Do not report an error or exit when a Tombstone store exists in the cluster [#4223](#)
- TiDB Binlog
 - Update the DDL binlog replication plan to guarantee the correctness of DDL event replication [#9304](#)

14.10.16 TiDB 2.1.4 Release Notes

On February 15, 2019, TiDB 2.1.4 is released. The corresponding TiDB Ansible 2.1.4 is also released. Compared with TiDB 2.1.3, this release has greatly improved the stability, the SQL optimizer, statistics, and the execution engine.

14.10.16.1 TiDB

- SQL Optimizer/Executor
 - Fix the issue that the VALUES function does not handle the FLOAT type correctly [#9223](#)
 - Fix the wrong result issue when casting Float to String in some cases [#9227](#)
 - Fix the wrong result issue of the FORMAT function in some cases [#9235](#)
 - Fix the panic issue when handling the Join query in some cases [#9264](#)
 - Fix the issue that the VALUES function does not handle the ENUM type correctly [#9280](#)
 - Fix the wrong result issue of DATE_ADD/DATE_SUB in some cases [#9284](#)
- Server
 - Optimize the “reload privilege success” log and change it to the DEBUG level [#9274](#)
- DDL
 - Change `tidb_ddl_reorg_worker_cnt` and `tidb_ddl_reorg_batch_size` to global variables [#9134](#)
 - Fix the bug caused by adding an index to a generated column in some abnormal conditions [#9289](#)

14.10.16.2 TiKV

- Fix the duplicate write issue when closing TiKV [#4146](#)
- Fix the abnormal result issue of the event listener in some cases [#4132](#)

14.10.16.3 Tools

- Lightning
 - Optimize the memory usage [#107](#), [#108](#)
 - Remove the chunk separation of dump files to avoid an extra parsing of dump files [#109](#)
 - Limit the I/O concurrency of reading dump files, to avoid performance degradation caused by too many cache misses [#110](#)
 - Support importing data in batches for a single table, to improve import stability [#110](#)
 - Enable auto compactions in the import mode in TiKV [#4199](#)
 - Support disabling the TiKV periodic Level-1 compaction parameter, because the Level-1 compaction is automatically executed in the import mode when the TiKV cluster version is 2.1.4 or later [#119](#)
 - Limit the number of import engines to avoid consuming too much importer disk space [#119](#)
- Support splitting chunks using the TiDB statistics in sync-diff-inspector [#197](#)

14.10.17 TiDB 2.1.3 Release Notes

On January 28, 2019, TiDB 2.1.3 is released. The corresponding TiDB Ansible 2.1.3 is also released. Compared with TiDB 2.1.2, this release has great improvement in system stability, SQL optimizer, statistics information, and execution engine.

14.10.17.1 TiDB

- SQL Optimizer/Executor
 - Fix the panic issue of Prepared Plan Cache in some cases [#8826](#)
 - Fix the issue that Range computing is wrong when the index is a prefix index [#8851](#)
 - Make `CAST(str AS TIME(N))` return null if the string is in the illegal TIME format when `SQL_MODE` is not strict [#8966](#)
 - Fix the panic issue of Generated Column during the process of UPDATE in some cases [#8980](#)
 - Fix the upper bound overflow issue of the statistics histogram in some cases [#8989](#)
 - Support Range for `_tidb_rowid` construction queries, to avoid full table scan and reduce cluster stress [#9059](#)

- Return an error when the `CAST(AS TIME)` precision is too big [#9058](#)
- Allow using `Sort Merge Join` in the Cartesian product [#9037](#)
- Fix the issue that the statistics worker cannot resume after the panic in some cases [#9085](#)
- Fix the issue that `Sort Merge Join` returns the wrong result in some cases [#9046](#)
- Support returning the JSON type in the `CASE` clause [#8355](#)
- Server
 - Return a warning instead of an error when the non-TiDB hint exists in the comment [#8766](#)
 - Verify the validity of the configured `TIMEZONE` value [#8879](#)
 - Optimize the `QueryDurationHistogram` metrics item to display more statement types [#8875](#)
 - Fix the lower bound overflow issue of bigint in some cases [#8544](#)
 - Support the `ALLOW_INVALID_DATES` SQL mode [#9110](#)
- DDL
 - Fix a `RENAME TABLE` compatibility issue to keep the behavior consistent with that of MySQL [#8808](#)
 - Support making concurrent changes of `ADD INDEX` take effect immediately [#8786](#)
 - Fix the `UPDATE` panic issue during the process of `ADD COLUMN` in some cases [#8906](#)
 - Fix the issue of concurrently creating Table Partition in some cases [#8902](#)
 - Support converting the `utf8` character set to `utf8mb4` [#8951](#) [#9152](#)
 - Fix the issue of Shard Bits overflow [#8976](#)
 - Support outputting the column character sets in `SHOW CREATE TABLE` [#9053](#)
 - Fix the issue of the maximum length limit of the varchar type column in `utf8mb4` [#8818](#)
 - Support `ALTER TABLE TRUNCATE TABLE PARTITION` [#9093](#)
 - Resolve the charset when the charset is not provided [#9147](#)

14.10.17.2 PD

- Fix the Watch issue related to leader election [#1396](#)

14.10.17.3 TiKV

- Support obtaining the monitoring information using the HTTP method [#3855](#)
- Fix the NULL issue of `data_format` [#4075](#)
- Add verifying the range for scan requests [#4124](#)

14.10.17.4 Tools

- TiDB Binlog

- Fix the no available pump issue while TiDB is started or restarted [#157](#)
- Enable outputting the Pump client log [#165](#)
- Fix the data inconsistency issue caused by the unique key containing the NULL value when the table only has the unique key and does not have the primary key

14.10.18 TiDB 2.1.2 Release Notes

On December 22, 2018, TiDB 2.1.2 is released. The corresponding TiDB Ansible 2.1.2 is also released. Compared with TiDB 2.1.1, this release has great improvement in system compatibility and stability.

14.10.18.1 TiDB

- Make TiDB compatible with TiDB Binlog of the Kafka version [#8747](#)
- Improve the exit mechanism of TiDB in a rolling update [#8707](#)
- Fix the panic issue caused by adding the index for the generated column in some cases [#8676](#)
- Fix the issue that the optimizer cannot find the optimal query plan when `TIDB_SMJ ↩ Hint` exists in the SQL statement in some cases [#8729](#)
- Fix the issue that `AntiSemiJoin` returns an incorrect result in some cases [#8730](#)
- Improve the valid character check of the `utf8` character set [#8754](#)
- Fix the issue that the field of the time type might return an incorrect result when the write operation is performed before the read operation in a transaction [#8746](#)

14.10.18.2 PD

- Fix the Region information update issue about Region merge [#1377](#)

14.10.18.3 TiKV

- Support the configuration format in the unit of DAY (d) and fix the configuration compatibility issue [#3931](#)
- Fix the possible panic issue caused by Approximate Size Split [#3942](#)
- Fix two issues about Region merge [#3822](#), [#3873](#)

14.10.18.4 Tools

- TiDB Lightning
 - Make TiDB 2.1.0 the minimum cluster version supported by Lightning
 - Fix the content error of the file involving parsed JSON data in Lightning [#144](#)
 - Fix the issue that `Too many open engines` occurs after the checkpoint is used to restart Lightning

- TiDB Binlog
 - Eliminate some bottlenecks of Drainer writing data to Kafka
 - Support the Kafka version of TiDB Binlog

14.10.19 TiDB 2.1.1 Release Notes

On December 12, 2018, TiDB 2.1.1 is released. Compared with TiDB 2.1.0, this release has great improvement in stability, SQL optimizer, statistics information, and execution engine.

14.10.19.1 TiDB

- SQL Optimizer/Executor
 - Fix the round error of the negative date [#8574](#)
 - Fix the issue that the `uncompress` function does not check the data length [#8606](#)
 - Reset bind arguments of the `prepare` statement after the `execute` command is executed [#8652](#)
 - Support automatically collecting the statistics information of a partition table [#8649](#)
 - Fix the wrongly configured integer type when pushing down the `abs` function [#8628](#)
 - Fix the data race on the JSON column [#8660](#)
- Server
 - Fix the issue that the transaction obtained TSO is incorrect when PD breaks down [#8567](#)
 - Fix the bootstrap failure caused by the statement that does not conform to ANSI standards [#8576](#)
 - Fix the issue that incorrect parameters are used in transaction retries [#8638](#)
- DDL
 - Change the default character set and collation of tables into `utf8mb4` [#8590](#)
 - Add the `ddl_reorg_batch_size` variable to control the speed of adding indexes [#8614](#)
 - Make the character set and collation options content in DDL case-insensitive [#8611](#)
 - Fix the issue of adding indexes for generated columns [#8655](#)

14.10.19.2 PD

- Fix the issue that some configuration items cannot be set to 0 in the configuration file [#1334](#)

- Check the undefined configuration when starting PD [#1362](#)
- Avoid transferring the leader to a newly created peer, to optimize the possible delay [#1339](#)
- Fix the issue that `RaftCluster` cannot stop caused by deadlock [#1370](#)

14.10.19.3 TiKV

- Avoid transferring the leader to a newly created peer, to optimize the possible delay [#3878](#)

14.10.19.4 Tools

- Lightning
 - Optimize the `analyze` mechanism on imported tables to increase the import speed
 - Support storing the checkpoint information to a local file
- TiDB Binlog
 - Fix the output bug of pb files that a table only with the primary key column cannot generate the pb event

14.10.20 TiDB 2.1 GA Release Notes

On November 30, 2018, TiDB 2.1 GA is released. See the following updates in this release. Compared with TiDB 2.0, this release has great improvements in stability, performance, compatibility, and usability.

14.10.20.1 TiDB

- SQL Optimizer
 - Optimize the selection range of `Index Join` to improve the execution performance
 - Optimize the selection of outer table for `Index Join` and use the table with smaller estimated value of Row Count as the outer table
 - Optimize Join Hint `TIDB_SMJ` so that Merge Join can be used even without proper index available
 - Optimize Join Hint `TIDB_INLJ` to specify the Inner table to Join
 - Optimize correlated subquery, push down Filter, and extend the index selection range, to improve the efficiency of some queries by orders of magnitude
 - Support using Index Hint and Join Hint in the `UPDATE` and `DELETE` statement
 - Support pushing down more functions: `ABS/CEIL/FLOOR/IS TRUE/IS FALSE`

- Optimize the constant folding algorithm for the `IF` and `IFNULL` built-in functions
- Optimize the output of the `EXPLAIN` statement and use hierarchy structure to show the relationship between operators
- SQL executor
 - Refactor all the aggregation functions and improve execution efficiency of the `Stream` and `Hash` aggregation operators
 - Implement the parallel `Hash Aggregate` operators and improve the computing performance by 350% in some scenarios
 - Implement the parallel `Project` operators and improve the performance by 74% in some scenarios
 - Read the data of the Inner table and Outer table of `Hash Join` concurrently to improve the execution performance
 - Optimize the execution speed of the `REPLACE INTO` statement and increase the performance nearly by 10 times
 - Optimize the memory usage of the time data type and decrease the memory usage of the time data type by fifty percent
 - Optimize the point select performance and improve the point select efficiency result of Sysbench by 60%
 - Improve the performance of TiDB on inserting or updating wide tables by 20 times
 - Support configuring the memory upper limit of a single statement in the configuration file
 - Optimize the execution of Hash Join, if the Join type is Inner Join or Semi Join and the inner table is empty, return the result without reading data from the outer table
 - Support using the `EXPLAIN ANALYZE` statement to check the runtime statistics including the execution time and the number of returned rows of each operator
- Statistics
 - Support enabling auto ANALYZE statistics only during certain period of the day
 - Support updating the table statistics automatically according to the feedback of the queries
 - Support configuring the number of buckets in the histogram using the `ANALYZE ↳ TABLE WITH BUCKETS` statement
 - Optimize the Row Count estimation algorithm using histogram for mixed queries of equality query and range queries
- Expressions

- Support following built-in function:

- * `json_contains`
- * `json_contains_path`
- * `encode/decode`

- Server

- Support queuing the locally conflicted transactions within tidb-server instance to optimize the performance of conflicted transactions
- Support Server Side Cursor
- Add the [HTTP API](#)
 - * Scatter the distribution of table Regions in the TiKV cluster
 - * Control whether to open the `general log`
 - * Support modifying the log level online
 - * Check the TiDB cluster information
- Add the `auto_analyze_ratio` system variables to control the ratio of Analyze
- Add the `tidb_retry_limit` system variable to control the automatic retry times of transactions
- Add the `tidb_disable_txn_auto_retry` system variable to control whether the transaction retries automatically
- Support using `admin show slow` statement to obtain the slow queries
- Add the `tidb_slow_log_threshold` environment variable to set the threshold of slow log automatically
- Add the `tidb_query_log_max_len` environment variable to set the length of the SQL statement to be truncated in the log dynamically

- DDL

- Support the parallel execution of the Add index statement and other statements to avoid the time consuming Add index operation blocking other operations
- Optimize the execution speed of `ADD INDEX` and improve it greatly in some scenarios
- Support the `select tidb_is_ddl_owner()` statement to facilitate deciding whether TiDB is DDL Owner
- Support the `ALTER TABLE FORCE` syntax
- Support the `ALTER TABLE RENAME KEY TO` syntax
- Add the table name and database name in the output information of `admin show → ddl jobs`

- Support using the `ddl/owner/resign` HTTP interface to release the DDL owner and start electing a new DDL owner
- Compatibility
 - Support more MySQL syntaxes
 - Make the `BIT` aggregate function support the `ALL` parameter
 - Support the `SHOW PRIVILEGES` statement
 - Support the `CHARACTER SET` syntax in the `LOAD DATA` statement
 - Support the `IDENTIFIED WITH` syntax in the `CREATE USER` statement
 - Support the `LOAD DATA IGNORE LINES` statement
 - The `Show ProcessList` statement returns more accurate information

14.10.20.2 Placement Driver (PD)

- Optimize availability
 - Introduce the version control mechanism and support rolling update of the cluster compatibly
 - **Enable Raft PreVote** among PD nodes to avoid leader reelection when network recovers after network isolation
 - Enable `raft learner` by default to lower the risk of unavailable data caused by machine failure during scheduling
 - TSO allocation is no longer affected by the system clock going backwards
 - Support the `Region merge` feature to reduce the overhead brought by metadata
- Optimize the scheduler
 - Optimize the processing of Down Store to speed up making up replicas
 - Optimize the hotspot scheduler to improve its adaptability when traffic statistics information jitters
 - Optimize the start of Coordinator to reduce the unnecessary scheduling caused by restarting PD
 - Optimize the issue that Balance Scheduler schedules small Regions frequently
 - Optimize Region merge to consider the number of rows within the Region
 - **Add more commands to control the scheduling policy**
 - Improve `PD simulator` to simulate the scheduling scenarios
- API and operation tools

- Add the `GetPrevRegion` interface to support the TiDB `reverse scan` feature
 - Add the `BatchSplitRegion` interface to speed up TiKV Region splitting
 - Add the `GCSafePoint` interface to support distributed GC in TiDB
 - Add the `GetAllStores` interface, to support distributed GC in TiDB
 - pd-ctl supports:
 - * using statistics for Region split
 - * calling `jq` to format the JSON output
 - * checking the Region information of the specified store
 - * checking topN Region list sorted by versions
 - * checking topN Region list sorted by size
 - * more precise TSO encoding
 - `pd-recover` doesn't need to provide the `max-replica` parameter
- Metrics
 - Add related metrics for `Filter`
 - Add metrics about etcd Raft state machine
 - Performance
 - Optimize the performance of Region heartbeat to reduce the memory overhead brought by heartbeats
 - Optimize the Region tree performance
 - Optimize the performance of computing hotspot statistics

14.10.20.3 TiKV

- Coprocessor
 - Add more built-in functions
 - Add Coprocessor `ReadPool` to improve the concurrency in processing the requests
 - Fix the time function parsing issue and the time zone related issues
 - Optimize the memory usage for pushdown aggregation computing
- Transaction
 - Optimize the read logic and memory usage of MVCC to improve the performance of the scan operation and the performance of full table scan is 1 time better than that in TiDB 2.0
 - Fold the continuous Rollback records to ensure the read performance

- Add the `UnsafeDestroyRange` API to support collecting space for the dropping table/index
- Separate the GC module to reduce the impact on write
- Add the `upper` bound support in the `kv_scan` command
- Raftstore
 - Improve the snapshot writing process to avoid RocksDB stall
 - Add the `LocalReader` thread to process read requests and reduce the delay for read requests
 - Support `BatchSplit` to avoid large Region brought by large amounts of write
 - Support `Region Split` according to statistics to reduce the I/O overhead
 - Support `Region Split` according to the number of keys to improve the concurrency of index scan
 - Improve the Raft message process to avoid unnecessary delay brought by `Region → Split`
 - Enable the `PreVote` feature by default to reduce the impact of network isolation on services
- Storage Engine
 - Fix the `CompactFile` bug in RocksDB and reduce the impact on importing data using Lightning
 - Upgrade RocksDB to v5.15 to fix the possible issue of snapshot file corruption
 - Improve `IngestExternalFile` to avoid the issue that flush could block write
- tikv-ctl
 - Add the `ldb` command to diagnose RocksDB related issues
 - The `compact` command supports specifying whether to compact data in the bottommost level

14.10.20.4 Tools

- Fast full import of large amounts of data: [TiDB Lightning](#)
- Support new [TiDB Binlog](#)

14.10.20.5 Upgrade caveat

- TiDB 2.1 does not support downgrading to v2.0.x or earlier due to the adoption of the new storage engine
- Parallel DDL is enabled in TiDB 2.1, so the clusters with TiDB version earlier than 2.0.1 cannot upgrade to 2.1 using rolling update. You can choose either of the following two options:
 - Stop the cluster and upgrade to 2.1 directly
 - Roll update to 2.0.1 or later 2.0.x versions, and then roll update to the 2.1 version
- If you upgrade from TiDB 2.0.6 or earlier to TiDB 2.1, check if there is any ongoing DDL operation, especially the time consuming `Add Index` operation, because the DDL operations slow down the upgrading process. If there is ongoing DDL operation, wait for the DDL operation finishes and then roll update.

14.10.21 TiDB 2.1 RC5 Release Notes

On November 12, 2018, TiDB 2.1 RC5 is released. Compared with TiDB 2.1 RC4, this release has great improvement in stability, SQL optimizer, statistics information, and execution engine.

14.10.21.1 TiDB

- SQL Optimizer
 - Fix the issue that `IndexReader` reads the wrong handle in some cases [#8132](#)
 - Fix the issue occurred while the `IndexScan Prepared` statement uses `Plan Cache` [#8055](#)
 - Fix the issue that the result of the `Union` statement is unstable [#8165](#)
- SQL Execution Engine
 - Improve the performance of TiDB on inserting or updating wide tables [#8024](#)
 - Support the unsigned int flag in the `Truncate` built-in function [#8068](#)
 - Fix the error occurred while converting JSON data to the decimal type [#8109](#)
 - Fix the error occurred when you `Update` the float type [#8170](#)
- Statistics
 - Fix the incorrect statistics issue during point queries in some cases [#8035](#)
 - Fix the selectivity estimation of statistics for primary key in some cases [#8149](#)
 - Fix the issue that the statistics of deleted tables are not cleared up for a long period of time [#8182](#)
- Server

- Improve the readability of logs and make logs better
 - * [#8063](#)
 - * [#8053](#)
 - * [#8224](#)
- Fix the error occurred when obtaining the table data of `infoschema.profiling` [#8096](#)
- Replace the unix socket with the pumps client to write binlogs [#8098](#)
- Add the threshold value for the `tidb_slow_log_threshold` environment variable, which dynamically sets the slow log [#8094](#)
- Add the original length of a SQL statement truncated while the `tidb_query_log_max_len` environment variable dynamically sets logs [#8200](#)
- Add the `tidb_opt_write_row_id` environment variable to control whether to allow writing `_tidb_rowid` [#8218](#)
- Add an upper bound to the `Scan` command of ticlient, to avoid overbound scan [#8081](#), [#8247](#)
- DDL
 - Fix the issue that executing DDL statements in transactions encounters an error in some cases [#8056](#)
 - Fix the issue that executing `truncate table` in partition tables does not take effect [#8103](#)
 - Fix the issue that the DDL operation does not roll back correctly after being cancelled in some cases [#8057](#)
 - Add the `admin show next_row_id` command to return the next available row ID [#8268](#)

14.10.21.2 PD

- Fix the issues related to `pd-ctl` reading the Region key
 - [#1298](#)
 - [#1299](#)
 - [#1308](#)
- Fix the issue that the `regions/check` API returns the wrong result [#1311](#)
- Fix the issue that PD cannot restart join after a PD join failure [#1279](#)
- Fix the issue that `watch leader` might lose events in some cases [#1317](#)

14.10.21.3 TiKV

- Improve the error message of `WriteConflict` [#3750](#)
- Add the panic mark file [#3746](#)
- Downgrade grpcio to avoid the segment fault issue caused by the new version of gRPC [#3650](#)
- Add an upper limit to the `kv_scan` interface [#3749](#)

14.10.21.4 Tools

- Support the TiDB-Binlog cluster, which is not compatible with the older version of binlog [#8093](#), [documentation](#)

14.10.22 TiDB 2.1 RC4 Release Notes

On October 23, 2018, TiDB 2.1 RC4 is released. Compared with TiDB 2.1 RC3, this release has great improvement in stability, SQL optimizer, statistics information, and execution engine.

14.10.22.1 TiDB

- SQL Optimizer
 - Fix the issue that column pruning of `UnionAll` is incorrect in some cases [#7941](#)
 - Fix the issue that the result of the `UnionAll` operator is incorrect in some cases [#8007](#)
- SQL Execution Engine
 - Fix the precision issue of the `AVG` function [#7874](#)
 - Support using the `EXPLAIN ANALYZE` statement to check the runtime statistics including the execution time and the number of returned rows of each operator during the query execution process [#7925](#)
 - Fix the panic issue of the `PointGet` operator when a column of a table appears multiple times in the result set [#7943](#)
 - Fix the panic issue caused by too large values in the `Limit` subclause [#8002](#)
 - Fix the panic issue during the execution process of the `AddDate/SubDate` statement in some cases [#8009](#)
- Statistics
 - Fix the issue of judging the prefix of the histogram low-bound of the combined index as out of range [#7856](#)
 - Fix the memory leak issue caused by statistics collecting [#7873](#)
 - Fix the panic issue when the histogram is empty [#7928](#)
 - Fix the issue that the histogram bound is out of range when the statistics is being uploaded [#7944](#)
 - Limit the maximum length of values in the statistics sampling process [#7982](#)
- Server
 - Refactor Latch to avoid misjudgment of transaction conflicts and improve the execution performance of concurrent transactions [#7711](#)
 - Fix the panic issue caused by collecting slow queries in some cases [#7874](#)

- Fix the panic issue when ESCAPED BY is an empty string in the LOAD DATA statement [#8005](#)
- Complete the “coprocessor error” log information [#8006](#)
- Compatibility
 - Set the Command field of the SHOW PROCESSLIST result to Sleep when the query is empty [#7839](#)
- Expressions
 - Fix the constant folding issue of the SYSDATE function [#7895](#)
 - Fix the issue that SUBSTRING_INDEX panics in some cases [#7897](#)
- DDL
 - Fix the stack overflow issue caused by throwing the invalid ddl job type error [#7958](#)
 - Fix the issue that the result of ADMIN CHECK TABLE is incorrect in some cases [#7975](#)

14.10.22.2 PD

- Fix the issue that the tombstone TiKV is not removed from Grafana [#1261](#)
- Fix the data race issue when grpc-go configures the status [#1265](#)
- Fix the issue that the PD server gets stuck caused by etcd startup failure [#1267](#)
- Fix the issue that data race might occur during leader switching [#1273](#)
- Fix the issue that extra warning logs might be output when TiKV becomes tombstone [#1280](#)

14.10.22.3 TiKV

- Optimize the RocksDB Write stall issue caused by applying snapshots [#3606](#)
- Add raftstore tick metrics [#3657](#)
- Upgrade RocksDB and fix the Write block issue and that the source file might be damaged by the Write operation when performing IngestExternalFile [#3661](#)
- Upgrade grpcio and fix the issue that “too many pings” is wrongly reported [#3650](#)

14.10.23 TiDB 2.1 RC3 Release Notes

On September 29, 2018, TiDB 2.1 RC3 is released. Compared with TiDB 2.1 RC2, this release has great improvement in stability, compatibility, SQL optimizer, and execution engine.

14.10.23.1 TiDB

- SQL Optimizer
 - Fix the incorrect result issue when a statement contains embedded LEFT OUTER
 ↪ JOIN #7689
 - Enhance the optimization rule of predicate pushdown on the JOIN statement
 #7645
 - Fix the optimization rule of predicate pushdown for the UnionScan operator
 #7695
 - Fix the issue that the unique key property of the Union operator is not correctly set
 #7680
 - Enhance the optimization rule of constant folding #7696
 - Optimize the data source in which the filter is null after propagation to table dual
 #7756
- SQL Execution Engine
 - Optimize the performance of read requests in a transaction #7717
 - Optimize the cost of allocating Chunk memory in some executors #7540
 - Fix the “index out of range” panic caused by the columns where point queries get all NULL values #7790
- Server
 - Fix the issue that the memory quota in the configuration file does not take effect
 #7729
 - Add the `tidb_force_priority` system variable to set the execution priority for each statement #7694
 - Support using the `admin show slow` statement to obtain the slow query log
 #7785
- Compatibility
 - Fix the issue that the result of `charset/collation` is incorrect in `information_schema`
 ↪ `.schemas` #7751
 - Fix the issue that the value of the `hostname` system variable is empty #7750
- Expressions
 - Support the `init_vector` argument in the `AES_ENCRYPT/AES_DECRYPT` built-in function #7425
 - Fix the issue that the result of `Format` is incorrect in some expressions #7770
 - Support the `JSON_LENGTH` built-in function #7739
 - Fix the incorrect result issue when casting the unsigned integer type to the decimal type #7792
- DML

- Fix the issue that the result of the `INSERT ... ON DUPLICATE KEY UPDATE` statement is incorrect while updating the unique key [#7675](#)
- DDL
 - Fix the issue that the index value is not converted between time zones when you create a new index on a new column of the timestamp type [#7724](#)
 - Support appending new values for the enum type [#7767](#)
 - Support creating an etcd session quickly, which improves the cluster availability after network isolation [#7774](#)

14.10.23.2 PD

- New feature
 - Add the API to get the Region list by size in reverse order [#1254](#)
- Improvement
 - Return more detailed information in the Region API [#1252](#)
- Bug fix
 - Fix the issue that `adjacent-region-scheduler` might lead to a crash after PD switches the leader [#1250](#)

14.10.23.3 TiKV

- Performance
 - Optimize the concurrency for coprocessor requests [#3515](#)
- New features
 - Add the support for Log functions [#3603](#)
 - Add the support for the `sha1` function [#3612](#)
 - Add the support for the `truncate_int` function [#3532](#)
 - Add the support for the `year` function [#3622](#)
 - Add the support for the `truncate_real` function [#3633](#)
- Bug fixes
 - Fix the reporting error behavior related to time functions [#3487](#), [#3615](#)
 - Fix the issue that the time parsed from string is inconsistent with that in TiDB [#3589](#)

14.10.24 TiDB 2.1 RC2 Release Notes

On September 14, 2018, TiDB 2.1 RC2 is released. Compared with TiDB 2.1 RC1, this release has great improvement in stability, SQL optimizer, statistics information, and execution engine.

14.10.24.1 TiDB

- SQL Optimizer
 - Put forward a proposal of the next generation Planner [#7543](#)
 - Improve the optimization rules of constant propagation [#7276](#)
 - Enhance the computing logic of Range to enable it to handle multiple IN or EQUAL conditions simultaneously [#7577](#)
 - Fix the issue that the estimation result of TableScan is incorrect when Range is empty [#7583](#)
 - Support the PointGet operator for the UPDATE statement [#7586](#)
 - Fix the panic issue during the process of executing the FirstRow aggregate function in some conditions [#7624](#)
- SQL Execution Engine
 - Fix the potential DataRace issue when the HashJoin operator encounters an error [#7554](#)
 - Make the HashJoin operator read the inner table and build the hash table simultaneously [#7544](#)
 - Optimize the performance of Hash aggregate operators [#7541](#)
 - Optimize the performance of Join operators [#7493](#), [#7433](#)
 - Fix the issue that the result of UPDATE JOIN is incorrect when the Join order is changed [#7571](#)
 - Improve the performance of Chunk's iterator [#7585](#)
- Statistics
 - Fix the issue that the auto Analyze work repeatedly analyzes the statistics [#7550](#)
 - Fix the statistics update error that occurs when there is no statistics change [#7530](#)
 - Use the RC isolation level and low priority when building Analyze requests [#7496](#)
 - Support enabling statistics auto-analyze on certain period of a day [#7570](#)
 - Fix the panic issue when logging the statistics information [#7588](#)
 - Support configuring the number of buckets in the histogram using the ANALYZE → TABLE WITH BUCKETS statement [#7619](#)
 - Fix the panic issue when updating an empty histogram [#7640](#)
 - Update information_schema.tables.data_length using the statistics information [#7657](#)
- Server
 - Add Trace related dependencies [#7532](#)
 - Enable the mutex profile feature of Golang [#7512](#)
 - The Admin statement requires the Super_priv privilege [#7486](#)
 - Forbid users to Drop crucial system tables [#7471](#)
 - Switch from juju/errors to pkg/errors [#7151](#)
 - Complete the functional prototype of SQL Tracing [#7016](#)

- Remove the goroutine pool [#7564](#)
- Support viewing the goroutine information using the `USER1` signal [#7587](#)
- Set the internal SQL to high priority while TiDB is started [#7616](#)
- Use different labels to filter internal SQL and user SQL in monitoring metrics [#7631](#)
- Store the top 30 slow queries in the last week to the TiDB server [#7646](#)
- Put forward a proposal of setting the global system time zone for the TiDB cluster [#7656](#)
- Enrich the error message of “GC life time is shorter than transaction duration” [#7658](#)
- Set the global system time zone when starting the TiDB cluster [#7638](#)
- Compatibility
 - Add the unsigned flag for the `Year` type [#7542](#)
 - Fix the issue of configuring the result length of the `Year` type in the `Prepare` \rightarrow `/Execute` mode [#7525](#)
 - Fix the issue of inserting zero timestamp in the `Prepare/Execute` mode [#7506](#)
 - Fix the error handling issue of the integer division [#7492](#)
 - Fix the compatibility issue when processing `ComStmtSendLongData` [#7485](#)
 - Fix the error handling issue during the process of converting string to integer [#7483](#)
 - Optimize the accuracy of values in the `information_schema.columns_in_table` table [#7463](#)
 - Fix the compatibility issue when writing or updating the string type of data using the MariaDB client [#7573](#)
 - Fix the compatibility issue of aliases of the returned value [#7600](#)
 - Fix the issue that the `NUMERIC_SCALE` value of the float type is incorrect in the `information_schema.COLUMNS` table [#7602](#)
 - Fix the issue that Parser reports an error when the single line comment is empty [#7612](#)
- Expressions
 - Check the value of `max_allowed_packet` in the `insert` function [#7528](#)
 - Support the built-in function `json_contains` [#7443](#)
 - Support the built-in function `json_contains_path` [#7596](#)
 - Support the built-in function `encode/decode` [#7622](#)
 - Fix the issue that some time related functions are not compatible with the MySQL behaviors in some cases [#7636](#)
 - Fix the compatibility issue of parsing the time type of data in string [#7654](#)
 - Fix the issue that the time zone is not considered when computing the default value of the `DateTime` data [#7655](#)
- DML
 - Set correct `last_insert_id` in the `InsertOnDuplicateUpdate` statement [#7534](#)
 - Reduce the cases of updating the `auto_increment_id` counter [#7515](#)

- Optimize the error message of Duplicate Key [#7495](#)
 - Fix the `insert...select...on duplicate key update` issue [#7406](#)
 - Support the `LOAD DATA IGNORE LINES` statement [#7576](#)
- DDL
 - Add the DDL job type and the current schema version information in the monitor [#7472](#)
 - Complete the design of the `Admin Restore Table` feature [#7383](#)
 - Fix the issue that the default value of the `Bit` type exceeds 128 [#7249](#)
 - Fix the issue that the default value of the `Bit` type cannot be `NULL` [#7604](#)
 - Reduce the interval of checking `CREATE TABLE/DATABASE` in the DDL queue [#7608](#)
 - Use the `ddl/owner/resign` HTTP interface to release the DDL owner and start electing a new owner [#7649](#)
- TiKV Go Client
 - Support the issue that the `Seek` operation only obtains Key [#7419](#)
- Table Partition (Experimental)
 - Fix the issue that the `Bigint` type cannot be used as the partition key [#7520](#)
 - Support the rollback operation when an issue occurs during adding an index in the partitioned table [#7437](#)

14.10.24.2 PD

- Features
 - Support the `GetAllStores` interface [#1228](#)
 - Add the statistics of scheduling estimation in Simulator [#1218](#)
- Improvements
 - Optimize the handling process of down stores to make up replicas as soon as possible [#1222](#)
 - Optimize the start of Coordinator to reduce the unnecessary scheduling caused by restarting PD [#1225](#)
 - Optimize the memory usage to reduce the overhead caused by heartbeats [#1195](#)
 - Optimize error handling and improve the log information [#1227](#)
 - Support querying the Region information of a specific store in pd-ctl [#1231](#)
 - Support querying the topN Region information based on version comparison in pd-ctl [#1233](#)
 - Support more accurate TSO decoding in pd-ctl [#1242](#)
- Bug fix
 - Fix the issue that pd-ctl uses the `hot store` command to exit wrongly [#1244](#)

14.10.24.3 TiKV

- Performance
 - Support splitting Regions based on statistics estimation to reduce the I/O cost [#3511](#)
 - Reduce clone in the transaction scheduler [#3530](#)
- Improvements
 - Add the pushdown support for a large number of built-in functions
 - Add the `leader-transfer-max-log-lag` configuration to fix the failure issue of leader scheduling in specific scenarios [#3507](#)
 - Add the `max-open-engines` configuration to limit the number of engines opened by `tikv-importer` simultaneously [#3496](#)
 - Limit the cleanup speed of garbage data to reduce the impact on `snapshot apply` [#3547](#)
 - Broadcast the commit message for crucial Raft messages to avoid unnecessary delay [#3592](#)
- Bug fixes
 - Fix the leader election issue caused by discarding the `PreVote` message of the newly split Region [#3557](#)
 - Fix follower related statistics after merging Regions [#3573](#)
 - Fix the issue that the local reader uses obsolete Region information [#3565](#)

14.10.25 TiDB 2.1 RC1 Release Notes

On August 24, 2018, TiDB 2.1 RC1 is released! Compared with TiDB 2.1 Beta, this release has great improvement in stability, SQL optimizer, statistics information, and execution engine.

14.10.25.1 TiDB

- SQL Optimizer
 - Fix the issue that a wrong result is returned after the correlated subquery is decorrelated in some cases [#6972](#)
 - Optimize the output result of `Explain` [#7011](#) [#7041](#)
 - Optimize the choosing strategy of the outer table for `IndexJoin` [#7019](#)
 - Remove the Plan Cache of the non-PREPARE statement [#7040](#)
 - Fix the issue that the `INSERT` statement is not parsed and executed correctly in some cases [#7068](#)
 - Fix the issue that the `IndexJoin` result is not correct in some cases [#7150](#)
 - Fix the issue that the `NULL` value cannot be found using the unique index in some cases [#7163](#)

- Fix the range computing issue of the prefix index in UTF-8 [#7194](#)
- Fix the issue that result is not correct caused by eliminating the `Project` operator in some cases [#7257](#)
- Fix the issue that `USE INDEX(PRIMARY)` cannot be used when the primary key is an integer [#7316](#)
- Fix the issue that the index range cannot be computed using the correlated column in some cases [#7357](#)
- SQL Execution Engine
 - Fix the issue that the daylight saving time is not computed correctly in some cases [#6823](#)
 - Refactor the aggregation function framework to improve the execution efficiency of the `Stream` and `Hash` aggregation operators [#6852](#)
 - Fix the issue that the `Hash` aggregation operator cannot exit normally in some cases [#6982](#)
 - Fix the issue that `BIT_AND/BIT_OR/BIT_XOR` does not handle the non-integer data correctly [#6994](#)
 - Optimize the execution speed of the `REPLACE INTO` statement and increase the performance nearly 10 times [#7027](#)
 - Optimize the memory usage of time type data and decrease the memory usage of the time type data by fifty percent [#7043](#)
 - Fix the issue that the returned result is mixed with signed and unsigned integers in the `UNION` statement is not compatible with MySQL [#7112](#)
 - Fix the panic issue caused by the too much memory applied by `LPAD/RPAD`
→ `/TO_BASE64/FROM_BASE64/REPEAT` [#7171](#) [#7266](#) [#7409](#) [#7431](#)
 - Fix the incorrect result when `MergeJoin/IndexJoin` handles the `NUL` value [#7255](#)
 - Fix the incorrect result of `Outer Join` in some cases [#7288](#)
 - Improve the error message of `Data Truncated` to facilitate locating the wrong data and the corresponding field in the table [#7401](#)
 - Fix the incorrect result for `decimal` in some cases [#7001](#) [#7113](#) [#7202](#) [#7208](#)
 - Optimize the point select performance [#6937](#)
 - Prohibit the isolation level of `Read Committed` to avoid the underlying problem [#7211](#)
 - Fix the incorrect result of `LTRIM/RTRIM/TRIM` in some cases [#7291](#)
 - Fix the issue that the `MaxOneRow` operator cannot guarantee that the returned result does not exceed one row [#7375](#)
 - Divide the Coprocessor requests with too many ranges [#7454](#)
- Statistics
 - Optimize the mechanism of statistics dynamic collection [#6796](#)
 - Fix the issue that `Auto Analyze` does not work when data is updated frequently [#7022](#)
 - Decrease the Write conflicts during the statistics dynamic update process [#7124](#)
 - Optimize the cost estimation when the statistics is incorrect [#7175](#)

- Optimize the `AccessPath` cost estimation strategy [#7233](#)
- Server
 - Fix the bug in loading privilege information [#6976](#)
 - Fix the issue that the `Kill` command is too strict with privilege check [#6954](#)
 - Fix the issue of removing some binary numeric types [#6922](#)
 - Shorten the output log [#7029](#)
 - Handle the `mismatchClusterID` issue [#7053](#)
 - Add the `advertise-address` configuration item [#7078](#)
 - Add the `GrpcKeepAlive` option [#7100](#)
 - Add the connection or `Token` time monitor [#7110](#)
 - Optimize the data decoding performance [#7149](#)
 - Add the `PROCESSLIST` table in `INFORMMATION_SCHEMA` [#7236](#)
 - Fix the order issue when multiple rules are hit in verifying the privilege [#7211](#)
 - Change some default values of encoding related system variables to UTF-8 [#7198](#)
 - Make the slow query log show more detailed information [#7302](#)
 - Support registering tidb-server related information in PD and obtaining this information by HTTP API [#7082](#)
- Compatibility
 - Support Session variables `warning_count` and `error_count` [#6945](#)
 - Add `Scope` check when reading the system variables [#6958](#)
 - Support the `MAX_EXECUTION_TIME` syntax [#7012](#)
 - Support more statements of the `SET` syntax [#7020](#)
 - Add validity check when setting system variables [#7117](#)
 - Add the verification of the number of `PlaceHolders` in the `Prepare` statement [#7162](#)
 - Support `set character_set_results = null` [#7353](#)
 - Support the `flush status` syntax [#7369](#)
 - Fix the column size of `SET` and `ENUM` types in `information_schema` [#7347](#)
 - Support the `NATIONAL CHARACTER` syntax of statements for creating a table [#7378](#)
 - Support the `CHARACTER SET` syntax in the `LOAD DATA` statement [#7391](#)
 - Fix the column information of the `SET` and `ENUM` types [#7417](#)
 - Support the `IDENTIFIED WITH` syntax in the `CREATE USER` statement [#7402](#)
 - Fix the precision losing issue during `TIMESTAMP` computing process [#7418](#)
 - Support the validity verification of more `SYSTEM` variables [#7196](#)
 - Fix the incorrect result when the `CHAR_LENGTH` function computes the binary string [#7410](#)
 - Fix the incorrect `CONCAT` result in a statement involving `GROUP BY` [#7448](#)
 - Fix the imprecise type length issue when casting the `DECIMAL` type to the `STRING` type [#7451](#)
- DML
 - Fix the stability issue of the `Load Data` statement [#6927](#)
 - Fix the memory usage issue when performing some `Batch` operations [#7086](#)

- Improve the performance of the `Replace Into` statement [#7027](#)
- Fix the inconsistent precision issue when writing `CURRENT_TIMESTAMP` [#7355](#)
- DDL
 - Improve the method of DDL judging whether `Schema` is replicated to avoid mis-judgement in some cases [#7319](#)
 - Fix the `SHOW CREATE TABLE` result in adding index process [#6993](#)
 - Allow the default value of `text/blob/json` to be `NULL` in non-restrict `sql-mode` [#7230](#)
 - Fix the `ADD INDEX` issue in some cases [#7142](#)
 - Increase the speed of adding `UNIQUE-KEY` index operation largely [#7132](#)
 - Fix the truncating issue of the prefix index in UTF-8 character set [#7109](#)
 - Add the environment variable `tidb_ddl_reorg_priority` to control the priority of the `add-index` operation [#7116](#)
 - Fix the display issue of `AUTO-INCREMENT` in `information_schema.tables` [#7037](#)
 - Support the `admin show ddl jobs <number>` command and support output specified number of DDL jobs [#7028](#)
 - Support parallel DDL job execution [#6955](#)
- Table Partition (Experimental)
 - Support top level partition
 - Support Range Partition

14.10.25.2 PD

- Features
 - Introduce the version control mechanism and support rolling update of the cluster with compatibility
 - Enable the `region merge` feature
 - Support the `GetPrevRegion` interface
 - Support splitting Regions in batch
 - Support storing the GC safepoint
- Improvements
 - Optimize the issue that TSO allocation is affected by the system clock going backwards
 - Optimize the performance of handling Region heartbeats
 - Optimize the Region tree performance
 - Optimize the performance of computing hotspot statistics
 - Optimize returning the error code of API interface
 - Add options of controlling scheduling strategies
 - Prohibit using special characters in `label`
 - Improve the scheduling simulator

- Support splitting Regions using statistics in pd-ctl
- Support formatting JSON output by calling jq in pd-ctl
- Add metrics about etcd Raft state machine
- Bug fixes
 - Fix the issue that the namespace is not reloaded after switching Leader
 - Fix the issue that namespace scheduling exceeds the schedule limit
 - Fix the issue that hotspot scheduling exceeds the schedule limit
 - Fix the issue that wrong logs are output when the PD client closes
 - Fix the wrong statistics of Region heartbeat latency

14.10.25.3 TiKV

- Features
 - Support `batch split` to avoid too large Regions caused by the Write operation on hot Regions
 - Support splitting Regions based on the number of rows to improve the index scan efficiency
- Performance
 - Use `LocalReader` to separate the Read operation from the raftstore thread to lower the Read latency
 - Refactor the MVCC framework, optimize the memory usage and improve the scan Read performance
 - Support splitting Regions based on statistics estimation to reduce the I/O usage
 - Optimize the issue that the Read performance is affected by continuous Write operations on the rollback record
 - Reduce the memory usage of pushdown aggregation computing
- Improvements
 - Add the pushdown support for a large number of built-in functions and better charset support
 - Optimize the GC workflow, improve the GC speed and decrease the impact of GC on the system
 - Enable `prevote` to speed up service recovery when the network is abnormal
 - Add the related configuration items of RocksDB log files
 - Adjust the default configuration of `scheduler_latch`
 - Support setting whether to compact the data in the bottom layer of RocksDB when using tikv-ctl to compact data manually
 - Add the check for environment variables when starting TiKV
 - Support dynamically configuring the `dynamic_level_bytes` parameter based on the existing data
 - Support customizing the log format

- Integrate tikv-fail in tikv-ctl
- Add I/O metrics of threads
- Bug fixes
 - Fix decimal related issues
 - Fix the issue that gRPC `max_send_message_len` is set mistakenly
 - Fix the issue caused by misconfiguration of `region_size`

14.10.26 TiDB 2.1 Beta Release Notes

On June 29, 2018, TiDB 2.1 Beta is released! Compared with TiDB 2.0, this release has great improvement in stability, SQL optimizer, statistics information, and execution engine.

14.10.26.1 TiDB

- SQL Optimizer
 - Optimize the selection range of `Index Join` to improve the execution performance
 - Optimize correlated subquery, push down `Filter`, and extend the index range, to improve the efficiency of some queries by orders of magnitude
 - Support `Index Hint` and `Join Hint` in the `UPDATE` and `DELETE` statements
 - Validate Hint `TIDM_SMJ` when no available index exists
 - Support pushdown of the `ABS`, `CEIL`, `FLOOR`, `IS TRUE`, and `IS FALSE` functions
 - Handle the `IF` and `IFNULL` functions especially in the constant folding process
- SQL Execution Engine
 - Implement parallel `Hash Aggregate` operators and improve the computing performance of `Hash Aggregate` by 350% in some scenarios
 - Implement parallel `Project` operators and improve the performance by 74% in some scenarios
 - Read the data of the `Inner` table and `Outer` table of `Hash Join` concurrently to improve the execution performance
 - Fix incorrect results of `INSERT ... ON DUPLICATE KEY UPDATE ...` in some scenarios
 - Fix incorrect results of the `CONCAT_WS`, `FLOOR`, `CEIL`, and `DIV` built-in functions
- Server
 - Add the HTTP API to scatter the distribution of table Regions in the TiKV cluster
 - Add the `auto_analyze_ratio` system variable to control the threshold value of automatic `Analyze`
 - Add the HTTP API to control whether to open the general log
 - Add the HTTP API to modify the log level online
 - Add the user information in the general log and the slow query log

- Support the server side cursor
- Compatibility
 - Support more MySQL syntax
 - Make the `bit` aggregate function support the `ALL` parameter
 - Support the `SHOW PRIVILEGES` statement
- DML
 - Decrease the memory usage of the `INSERT INTO SELECT` statement
 - Fix the performance issue of `PlanCache`
 - Add the `tidb_retry_limit` system variable to control the automatic retry times of transactions
 - Add the `tidb_disable_txn_auto_retry` system variable to control whether the transaction tries automatically
 - Fix the accuracy issue of the written data of the `time` type
 - Support the queue of locally conflicted transactions to optimize the conflicted transaction performance
 - Fix `Affected Rows` of the `UPDATE` statement
 - Optimize the statement performance of `insert ignore on duplicate key`
 - ↪ `update`
- DDL
 - Optimize the execution speed of the `CreateTable` statement
 - Optimize the execution speed of `ADD INDEX` and improve it greatly in some scenarios
 - Fix the issue that the number of added columns by `Alter table add column` exceeds the limit of the number of table columns
 - Fix the issue that DDL job retries lead to an increasing pressure on TiKV in abnormal conditions
 - Fix the issue that TiDB continuously reloads the schema information in abnormal conditions
 - Do not output the `FOREIGN KEY` related information in the result of `SHOW CREATE TABLE`
 - Support the `select tidb_is_ddl_owner()` statement to facilitate judging whether TiDB is DDL Owner
 - Fix the issue that the index is deleted in the `Year` type in some scenarios
 - Fix the renaming table issue in the concurrent execution scenario
 - Support the `AlterTableForce` syntax
 - Support the `AlterTableRenameIndex` syntax with `FromKey` and `ToKey`
 - Add the table name and database name in the output information of `admin show ddl jobs`

14.10.26.2 PD

- Enable Raft PreVote between PD nodes to avoid leader reelection when network recovers after network isolation
- Optimize the issue that Balance Scheduler schedules small Regions frequently
- Optimize the hotspot scheduler to improve its adaptability in traffic statistics information jitters
- Skip the Regions with a large number of rows when scheduling `region merge`
- Enable `raft learner` by default to lower the risk of unavailable data caused by machine failure during scheduling
- Remove `max-replica` from `pd-recover`
- Add `Filter` metrics
- Fix the issue that Region information is not updated after tikv-ctl unsafe recovery
- Fix the issue that TiKV disk space is used up caused by replica migration in some scenarios
- Compatibility notes
 - Do not support rolling back to v2.0.x or earlier due to update of the new version storage engine
 - Enable `raft learner` by default in the new version of PD. If the cluster is upgraded from 1.x to 2.1, the machine should be stopped before upgrade or a rolling update should be first applied to TiKV and then PD

14.10.26.3 TiKV

- Upgrade Rust to the `nightly-2018-06-14` version
- Enable Raft PreVote to avoid leader reelection generated when network recovers after network isolation
- Add a metric to display the number of files and `ingest` related information in each layer of RocksDB
- Print `key` with too many versions when GC works
- Use `static metric` to optimize multi-label metric performance (YCSB `raw get` is improved by 3%)
- Remove `box` in multiple modules and use patterns to improve the operating performance (YCSB `raw get` is improved by 3%)
- Use `asynchronous log` to improve the performance of writing logs
- Add a metric to collect the thread status
- Decrease memory copy times by decreasing `box` used in the application to improve the performance

14.11 v2.0

14.11.1 TiDB 2.0.11 Release Notes

On January 03, 2019, TiDB 2.0.11 is released. The corresponding TiDB Ansible 2.0.11 is also released. Compared with TiDB 2.0.10, this release has great improvement in system compatibility and stability.

14.11.1.1 TiDB

- Fix the issue that the error is not handled properly when PD is in an abnormal condition [#8764](#)
- Fix the issue that the `Rename` operation on a table in TiDB is not compatible with that in MySQL [#8809](#)
- Fix the issue that the error message is wrongly reported when the `ADMIN CHECK TABLE` operation is performed in the process of executing the `ADD INDEX` statement [#8750](#)
- Fix the issue that the prefix index range is incorrect in some cases [#8877](#)
- Fix the panic issue of the `UPDATE` statement when columns are added in some cases [#8904](#)

14.11.1.2 TiKV

- Fix two issues about Region merge [#4003](#), [#4004](#)

14.11.2 TiDB 2.0.10 Release Notes

On December 18, 2018, TiDB 2.0.10 is released. The corresponding TiDB Ansible 2.0.10 is also released. Compared with TiDB 2.0.9, this release has great improvement in system compatibility and stability.

14.11.2.1 TiDB

- Fix the possible issue caused by canceling a DDL job [#8513](#)
- Fix the issue that the `ORDER BY` and `UNION` clauses cannot quote the column including a table name [#8514](#)
- Fix the issue that the `UNCOMPRESS` function does not judge the incorrect input length [#8607](#)
- Fix the issue encountered by `ANSI_QUOTES SQL_MODE` when upgrading TiDB [#8575](#)
- Fix the issue that `select` returns the wrong result in some cases [#8570](#)
- Fix the possible issue that TiDB cannot exit when it receives the exit signal [#8501](#)
- Fix the issue that `IndexLookUpJoin` returns the wrong result in some cases [#8508](#)
- Avoid pushing down the filter containing `GetVar` or `SetVar` [#8454](#)
- Fix the issue that the result length of the `UNION` clauses is incorrect in some cases [#8491](#)
- Fix the issue of `PREPARE FROM @var_name` [#8488](#)
- Fix the panic issue when dumping statistics information in some cases [#8464](#)
- Fix the statistics estimation issue of point queries in some cases [#8493](#)
- Fix the panic issue when the returned default `enum` value is a string [#8476](#)
- Fix the issue that too much memory is consumed in the scenario of wide tables [#8467](#)
- Fix the issue encountered when Parser incorrectly formats the mod opcode [#8431](#)

- Fix the panic issue caused by adding foreign key constraints in some cases [#8421](#), [#8410](#)
- Fix the issue that the YEAR column type incorrectly converts the zero value [#8396](#)
- Fix the panic issue occurred when the argument of the VALUES function is not a column [#8404](#)
- Disable Plan Cache for statements containing subqueries [#8395](#)

14.11.2.2 PD

- Fix the possible issue that RaftCluster cannot stop caused by deadlock [#1370](#)

14.11.2.3 TiKV

- Avoid transferring the leader to a newly created peer, to optimize the possible delay [#3929](#)
- Fix redundant Region heartbeats [#3930](#)

14.11.3 TiDB 2.0.9 Release Notes

On November 19, 2018, TiDB 2.0.9 is released. Compared with TiDB 2.0.8, this release has great improvement in system compatibility and stability.

14.11.3.1 TiDB

- Fix the issue caused by the empty statistics histogram [#7927](#)
- Fix the panic issue of the UNION ALL statement in some cases [#7942](#)
- Fix the stack overflow issue caused by wrong DDL Jobs [#7959](#)
- Add the slow log for the Commit operation [#7983](#)
- Fix the panic issue caused by the too large Limit value [#8004](#)
- Support specifying the utf8mb4 character set in the USING clause [#8048](#)
- Make the TRUNCATE built-in function support parameters of unsigned integer type [#8069](#)
- Fix the selectivity estimation issue of the primary key for the statistics module in some cases [#8150](#)
- Add the Session variable to control whether _tidb_rowid is allowed to be written in [#8126](#)
- Fix the panic issue of PhysicalProjection in some cases [#8154](#)
- Fix the unstable results of the Union statement in some cases [#8168](#)
- Fix the issue that NULL is not returned by values in the non-Insert statement [#8179](#)
- Fix the issue that the statistics module cannot clear the outdated data in some cases [#8184](#)
- Make the maximum allowed running time for a transaction a configurable option [8209](#)
- Fix the wrong comparison algorithm of expression rewriter in some cases [#8288](#)

- Eliminate the extra columns generated by the `UNION ORDER BY` statement [#8307](#)
- Support the `admin show next_row_id` statement [#8274](#)
- Fix the escape issue of special characters in the `Show Create Table` statement [#8321](#)
- Fix the unexpected errors in the `UNION` statement in some cases [#8318](#)
- Fix the issue that canceling a DDL job causes no rollback of a schema in some cases [#8312](#)
- Change `tidb_max_chunk_size` to a global variable [#8333](#)
- Add an upper bound to the `Scan` command of `ticlient`, to avoid overbound scan [#8309](#) [#8310](#)

14.11.3.2 PD

- Fix the issue that the PD server gets stuck caused by etcd startup failure [#1267](#)
- Fix the issues related to `pd-ctl` reading the Region key [#1298](#) [#1299](#) [#1308](#)
- Fix the issue that the `regions/check` API returns the wrong result [#1311](#)
- Fix the issue that PD cannot restart join after a PD join failure [#1279](#)

14.11.3.3 TiKV

- Add the `end-key` limit to the `kv_scan` interface [#3749](#)
- Abandon the `max-tasks-xxx` configuration and add `max-tasks-per-worker-xxx` [#3093](#)
- Fix the `CompactFiles` issue in RocksDB [#3789](#)

14.11.4 TiDB 2.0.8 Release Notes

On October 16, 2018, TiDB 2.0.8 is released. Compared with TiDB 2.0.7, this release has great improvement in system compatibility and stability.

14.11.4.1 TiDB

- Improvement
 - Slow down the AUTO-ID increasing speed when the `Update` statement does not modify the corresponding AUTO-INCREMENT column [#7846](#)
- Bug fixes
 - Quickly create a new etcd session to recover the service when the PD leader goes down [#7810](#)
 - Fix the issue that the time zone is not considered when the default value of the `DateTime` type is calculated [#7672](#)
 - Fix the issue that `duplicate key update` inserts values incorrectly in some conditions [#7685](#)

- Fix the issue that the predicate conditions of UnionScan are not pushed down [#7726](#)
- Fix the issue that the time zone is not correctly handled when you add the `TIMESTAMP` index [#7812](#)
- Fix the memory leak issue caused by the statistics module in some conditions [#7864](#)
- Fix the issue that the results of `ANALYZE` cannot be obtained in some abnormal conditions [#7871](#)
- Do not fold the function `SYSDATE`, to ensure the returned results are correct [#7894](#)
- Fix the `substring_index` panic issue in some conditions [#7896](#)
- Fix the issue that `OUTER JOIN` is mistakenly converted to `INNER JOIN` in some conditions [#7899](#)

14.11.4.2 TiKV

- Bug fix
 - Fix the issue that the memory consumed by Raftstore `EntryCache` keeps increasing when a node goes down [#3529](#)

14.11.5 TiDB 2.0.7 Release Notes

On September 7, 2018, TiDB 2.0.7 is released. Compared with TiDB 2.0.6, this release has great improvement in system compatibility and stability.

14.11.5.1 TiDB

- New Feature
 - Add the `PROCESSLIST` table in `information_schema` [#7286](#)
- Improvement
 - Collect more details about SQL statement execution and output the information in the `SLOW QUERY` log [#7364](#)
 - Drop the partition information in `SHOW CREATE TABLE` [#7388](#)
 - Improve the execution efficiency of the `ANALYZE` statement by setting it to the RC isolation level and low priority [#7500](#)
 - Speed up adding a unique index [#7562](#)
 - Add an option of controlling the DDL concurrency [#7563](#)
- Bug Fixes
 - Fix the issue that `USE INDEX(PRIMARY)` cannot be used in a table whose primary key is an integer [#7298](#)

- Fix the issue that `Merge Join` and `Index Join` output incorrect results when the inner row is `NULL` [#7301](#)
- Fix the issue that `Join` outputs an incorrect result when the chunk size is set too small [#7315](#)
- Fix the panic issue caused by a statement of creating a table involving `range ↵ column` [#7379](#)
- Fix the issue that `admin check table` mistakenly reports an error of a time-type column [#7457](#)
- Fix the issue that the data with a default value `current_timestamp` cannot be queried using the `=` condition [#7467](#)
- Fix the issue that the zero-length parameter inserted by using the `ComStmtSendLongData` command is mistakenly parsed to `NULL` [#7508](#)
- Fix the issue that `auto analyze` is repeatedly executed in specific scenarios [#7556](#)
- Fix the issue that the parser cannot parse a single line comment ended with a newline character [#7635](#)

14.11.5.2 TiKV

- Improvement
 - Open the `dynamic-level-bytes` parameter in an empty cluster by default, to reduce space amplification
- Bug Fix
 - Update `approximate size` and `approximate keys count` of a Region after Region merging

14.11.6 TiDB 2.0.6 Release Notes

On August 6, 2018, TiDB 2.0.6 is released. Compared with TiDB 2.0.5, this release has great improvement in system compatibility and stability.

14.11.6.1 TiDB

- Improvements
 - Make “set system variable” log shorter to save disk space [#7031](#)
 - Record slow operations during the execution of `ADD INDEX` in the log, to make troubleshooting easier [#7083](#)
 - Reduce transaction conflicts when updating statistics [#7138](#)
 - Improve the accuracy of row count estimation when the values pending to be estimated exceeds the statistics range [#7185](#)
 - Choose the table with a smaller estimated row count as the outer table for `Index Join` to improve its execution efficiency [#7277](#)

- Add the recover mechanism for panics occurred during the execution of `ANALYZE → TABLE`, to avoid that the tidb-server is unavailable caused by abnormal behavior in the process of collecting statistics [#7228](#)
- Return `NULL` and the corresponding warning when the results of `RPAD/LPAD` exceed the value of the `max_allowed_packet` system variable, compatible with MySQL [#7244](#)
- Set the upper limit of placeholders count in the `PREPARE` statement to 65535, compatible with MySQL [#7250](#)
- Bug Fixes
 - Fix the issue that the `DROP USER` statement is incompatible with MySQL behavior in some cases [#7014](#)
 - Fix the issue that statements like `INSERT/LOAD DATA` meet OOM after opening `tidb_batch_insert` [#7092](#)
 - Fix the issue that the statistics fail to automatically update when the data of a table keeps updating [#7093](#)
 - Fix the issue that the firewall breaks inactive gRPC connections [#7099](#)
 - Fix the issue that prefix index returns a wrong result in some scenarios [#7126](#)
 - Fix the panic issue caused by outdated statistics in some scenarios [#7155](#)
 - Fix the issue that one piece of index data is missed after the `ADD INDEX` operation in some scenarios [#7156](#)
 - Fix the wrong result issue when querying `NULL` values using the unique index in some scenarios [#7172](#)
 - Fix the messy code issue of the `DECIMAL` multiplication result in some scenarios [#7212](#)
 - Fix the wrong result issue of `DECIMAL` modulo operation in some scenarios [#7245](#)
 - Fix the issue that the `UPDATE/DELETE` statement in a transaction returns a wrong result under some special sequence of statements [#7219](#)
 - Fix the panic issue of the `UNION ALL/UPDATE` statement during the process of building the execution plan in some scenarios [#7225](#)
 - Fix the issue that the range of prefix index is calculated incorrectly in some scenarios [#7231](#)
 - Fix the issue that the `LOAD DATA` statement fails to write the binlog in some scenarios [#7242](#)
 - Fix the wrong result issue of `SHOW CREATE TABLE` during the execution process of `ADD INDEX` in some scenarios [#7243](#)
 - Fix the issue that panic occurs when `Index Join` does not initialize timestamps in some scenarios [#7246](#)
 - Fix the false alarm issue when `ADMIN CHECK TABLE` mistakenly uses the timezone in the session [#7258](#)
 - Fix the issue that `ADMIN CLEANUP INDEX` does not clean up the index in some scenarios [#7265](#)
 - Disable the Read Committed isolation level [#7282](#)

14.11.6.2 TiKV

- Improvements
 - Enlarge scheduler's default slots to reduce false conflicts
 - Reduce continuous records of rollback transactions, to improve the Read performance when conflicts are extremely severe
 - Limit the size and number of RocksDB log files, to reduce unnecessary disk usage in long-running condition
- Bug Fixes
 - Fix the crash issue when converting the data type from string to decimal

14.11.7 TiDB 2.0.5 Release Notes

On July 6, 2018, TiDB 2.0.5 is released. Compared with TiDB 2.0.4, this release has great improvement in system compatibility and stability.

14.11.7.1 TiDB

- New Features
 - Add the `tidb_disable_txn_auto_retry` system variable which is used to disable the automatic retry of transactions [#6877](#)
- Improvements
 - Optimize the cost calculation of `Selection` to make the result more accurate [#6989](#)
 - Select the query condition that completely matches the unique index or the primary key as the query path directly [#6966](#)
 - Execute necessary cleanup when failing to start the service [#6964](#)
 - Handle \N as NULL in the `Load Data` statement [#6962](#)
 - Optimize the code structure of CBO [#6953](#)
 - Report the monitoring metrics earlier when starting the service [#6931](#)
 - Optimize the format of slow queries by removing the line breaks in SQL statements and adding user information [#6920](#)
 - Support multiple asterisks in comments [#6858](#)
- Bug Fixes
 - Fix the issue that `KILL QUERY` always requires SUPER privilege [#7003](#)
 - Fix the issue that users might fail to login when the number of users exceeds 1024 [#6986](#)
 - Fix an issue about inserting unsigned `float/double` data [#6940](#)
 - Fix the compatibility of the `COM_FIELD_LIST` command to resolve the panic issue in some MariaDB clients [#6929](#)
 - Fix the `CREATE TABLE IF NOT EXISTS LIKE` behavior [#6928](#)
 - Fix an issue in the process of TopN pushdown [#6923](#)
 - Fix the ID record issue of the currently processing row when an error occurs in executing `Add Index` [#6903](#)

14.11.7.2 PD

- Fix the issue that replicas migration uses up TiKV disks space in some scenarios
- Fix the crash issue caused by `AdjacentRegionScheduler`

14.11.7.3 TiKV

- Fix the potential overflow issue in decimal operations
- Fix the dirty read issue that might occur in the process of merging

14.11.8 TiDB 2.0.4 Release Notes

On June 15, 2018, TiDB 2.0.4 is released. Compared with TiDB 2.0.3, this release has great improvement in system compatibility and stability.

14.11.8.1 TiDB

- Support the `ALTER TABLE t DROP COLUMN a CASCADE` syntax
- Support configuring the value of `tidb_snapshot` to TSO
- Refine the display of statement types in monitoring items
- Optimize the accuracy of query cost estimation
- Configure the `backoff max delay` parameter of gRPC
- Support configuring the memory threshold of a single statement in the configuration file
- Refactor the error of Optimizer
- Fix the side effects of the `Cast Decimal` data
- Fix the wrong result issue of the `Merge Join` operator in specific scenarios
- Fix the issue of converting the Null object to String
- Fix the issue of casting the JSON type of data to the JSON type
- Fix the issue that the result order is not consistent with MySQL in the condition of `Union + OrderBy`
- Fix the compliance rules issue when the `Union` statement checks the `Limit/OrderBy` clause
- Fix the compatibility issue of the `Union All` result
- Fix a bug in predicate pushdown
- Fix the compatibility issue of the `Union` statement with the `For Update` clause
- Fix the issue that the `concat_ws` function mistakenly truncates the result

14.11.8.2 PD

- Improve the behavior of the unset scheduling argument `max-pending-peer-count` by changing it to no limit for the maximum number of `PendingPeers`

14.11.8.3 TiKV

- Add the RocksDB PerfContext interface for debugging
- Remove the `import-mode` parameter
- Add the `region-properties` command for `tikv-ctl`
- Fix the issue that `reverse-seek` is slow when many RocksDB tombstones exist
- Fix the crash issue caused by `do_sub`
- Make GC record the log when GC encounters many versions of data

14.11.9 TiDB 2.0.3 Release Notes

On June 1, 2018, TiDB 2.0.3 is released. Compared with TiDB 2.0.2, this release has great improvement in system compatibility and stability.

14.11.9.1 TiDB

- Support modifying the log level online
- Support the `COM_CHANGE_USER` command
- Support using the `TIME` type parameters under the binary protocol
- Optimize the cost estimation of query conditions with the `BETWEEN` expression
- Do not display the `FOREIGN KEY` information in the result of `SHOW CREATE TABLE`
- Optimize the cost estimation for queries with the `LIMIT` clause
- Fix the issue about the `YEAR` type as the unique index
- Fix the issue about `ON DUPLICATE KEY UPDATE` in conditions without the unique index
- Fix the compatibility issue of the `CEIL` function
- Fix the accuracy issue of the `DIV` calculation in the `DECIMAL` type
- Fix the false alarm of `ADMIN CHECK TABLE`
- Fix the panic issue of `MAX/MIN` under specific expression parameters
- Fix the issue that the result of `JOIN` is null in special conditions
- Fix the `IN` expression issue when building and querying Range
- Fix a Range calculation issue when using `Prepare` to query and `Plan Cache` is enabled
- Fix the issue that the Schema information is frequently loaded in abnormal conditions

14.11.9.2 PD

- Fix the panic issue when collecting hot-cache metrics in specific conditions
- Fix the issue about scheduling of the obsolete Regions

14.11.9.3 TiKV

- Fix the bug that the learner flag mistakenly reports to PD
- Report an error instead of getting a result if divisor/dividend is 0 in `do_div_mod`

14.11.10 TiDB 2.0.2 Release Notes

On May 21, 2018, TiDB 2.0.2 is released. Compared with TiDB 2.0.1, this release has great improvement in system stability.

14.11.10.1 TiDB

- Fix the issue of pushing down the Decimal division expression
- Support using the `USE INDEX` syntax in the `Delete` statement
- Forbid using the `shard_row_id_bits` feature in columns with `Auto-Increment`
- Add the timeout mechanism for writing Binlog

14.11.10.2 PD

- Make the balance leader scheduler filter the disconnected nodes
- Modify the timeout of the transfer leader operator to 10s
- Fix the issue that the label scheduler does not schedule when the cluster Regions are in an unhealthy state
- Fix the improper scheduling issue of `evict leader scheduler`

14.11.10.3 TiKV

- Fix the issue that the Raft log is not printed
- Support configuring more gRPC related parameters
- Support configuring the timeout range of leader election
- Fix the issue that the obsolete learner is not deleted
- Fix the issue that the snapshot intermediate file is mistakenly deleted

14.11.11 TiDB 2.0.1 Release Notes

On May 16, 2018, TiDB 2.0.1 is released. Compared with TiDB 2.0.0 (GA), this release has great improvement in MySQL compatibility and system stability.

14.11.11.1 TiDB

- Update the progress of `Add Index` to the DDL job information in real time
- Add the `tidb_auto_analyze_ratio` session variable to control the threshold value of automatic statistics update
- Fix an issue that not all residual states are cleaned up when the transaction commit fails
- Fix a bug about adding indexes in some conditions

- Fix the correctness related issue when DDL modifies surface operations in some concurrent scenarios
- Fix a bug that the result of `LIMIT` is incorrect in some conditions
- Fix a capitalization issue of the `ADMIN CHECK INDEX` statement to make its index name case insensitive
- Fix a compatibility issue of the `UNION` statement
- Fix a compatibility issue when inserting data of `TIME` type
- Fix a goroutine leak issue caused by `copIteratorTaskSender` in some conditions
- Add an option for TiDB to control the behaviour of Binlog failure
- Refactor the `Coprocessor` slow log to distinguish between the scenario of tasks with long processing time and long waiting time
- Log nothing when meeting MySQL protocol handshake error, to avoid too many logs caused by the load balancer Keep Alive mechanism
- Refine the “Out of range value for column” error message
- Fix a bug when there is a subquery in an `Update` statement
- Change the behaviour of handling `SIGTERM`, and do not wait for all queries to terminate anymore

14.11.11.2 PD

- Add the `Scatter Range` scheduler to balance Regions with the specified key range
- Optimize the scheduling of Merge Region to prevent the newly split Region from being merged
- Add Learner related metrics
- Fix the issue that the scheduler is mistakenly deleted after restart
- Fix the error that occurs when parsing the configuration file
- Fix the issue that the etcd leader and the PD leader are not replicated
- Fix the issue that Learner still appears after it is closed
- Fix the issue that Regions fail to load because the packet size is too large

14.11.11.3 TiKV

- Fix the issue that `SELECT FOR UPDATE` prevents others from reading
- Optimize the slow query log
- Reduce the number of `thread_yield` calls
- Fix the bug that raftstore is accidentally blocked when generating the snapshot
- Fix the issue that Learner cannot be successfully elected in special conditions
- Fix the issue that split might cause dirty read in extreme conditions
- Correct the default value of the read thread pool configuration
- Speed up Delete Range

14.11.12 TiDB 2.0 Release Notes

On April 27, 2018, TiDB 2.0 GA is released! Compared with TiDB 1.0, this release has great improvement in MySQL compatibility, SQL optimizer, executor, and stability.

14.11.12.1 TiDB

- SQL Optimizer
 - Use more compact data structure to reduce the memory usage of statistics information
 - Speed up loading statistics information when starting a tidb-server process
 - Support updating statistics information dynamically **experimental**
 - Optimize the cost model to provide more accurate query cost evaluation
 - Use **Count-Min Sketch** to estimate the cost of point queries more accurately
 - Support analyzing more complex conditions to make full use of indexes
 - Support manually specifying the **Join** order using the **STRAIGHT_JOIN** syntax
 - Use the Stream Aggregation operator when the GROUP BY clause is empty to improve the performance
 - Support using indexes for the MAX/MIN function
 - Optimize the processing algorithms for correlated subqueries to support decorrelating more types of correlated subqueries and transform them to **Left Outer ↪ Join**
 - Extend **IndexLookupJoin** to be used in matching the index prefix
- SQL Execution Engine
 - Refactor all operators using the Chunk architecture, improve the execution performance of analytical queries, and reduce memory usage. There is a significant improvement in the TPC-H benchmark result.
 - Support the Streaming Aggregation operators pushdown
 - Optimize the **Insert Into Ignore** statement to improve the performance by over 10 times
 - Optimize the **Insert On Duplicate Key Update** statement to improve the performance by over 10 times
 - Optimize **Load Data** to improve the performance by over 10 times
 - Push down more data types and functions to TiKV
 - Support computing the memory usage of physical operators, and specifying the processing behavior in the configuration file and system variables when the memory usage exceeds the threshold
 - Support limiting the memory usage by a single SQL statement to reduce the risk of OOM
 - Support using implicit RowID in CRUD operations
 - Improve the performance of point queries
- Server
 - Support the Proxy Protocol
 - Add more monitoring metrics and refine the log
 - Support validating the configuration files
 - Support obtaining the information of TiDB parameters through HTTP API
 - Resolve Lock in the Batch mode to speed up garbage collection

- Support multi-threaded garbage collection
- Support TLS
- Compatibility
 - Support more MySQL syntaxes
 - Support modifying the `lower_case_table_names` system variable in the configuration file to support the OGG data replication tool
 - Improve compatibility with the Navicat management tool
 - Support displaying the table creating time in `Information_Schema`
 - Fix the issue that the return types of some functions/expressions differ from MySQL
 - Improve compatibility with JDBC
 - Support more SQL Modes
- DDL
 - Optimize the `Add Index` operation to greatly improve the execution speed in some scenarios
 - Attach a lower priority to the `Add Index` operation to reduce the impact on online business
 - Output more detailed status information of the DDL jobs in `Admin Show DDL`
 ↳ `Jobs`
 - Support querying the original statements of currently running DDL jobs using `Admin Show DDL Job Queries JobID`
 - Support recovering the index data using `Admin Recover Index` for disaster recovery
 - Support modifying Table Options using the `Alter` statement

14.11.12.2 PD

- Support `Region Merge`, to merge empty Regions after deleting data `experimental`
- Support `Raft Learner` `experimental`
- Optimize the scheduler
 - Make the scheduler to adapt to different Region sizes
 - Improve the priority and speed of restoring data during TiKV outage
 - Speed up data transferring when removing a TiKV node
 - Optimize the scheduling policies to prevent the disks from becoming full when the space of TiKV nodes is insufficient
 - Improve the scheduling efficiency of the balance-leader scheduler
 - Reduce the scheduling overhead of the balance-region scheduler
 - Optimize the execution efficiency of the hot-region scheduler
- Operations interface and configuration
 - Support TLS

- Support prioritizing the PD leaders
- Support configuring the scheduling policies based on labels
- Support configuring stores with a specific label not to schedule the Raft leader
- Support splitting Region manually to handle the hotspot in a single Region
- Support scattering a specified Region to manually adjust Region distribution in some cases
- Add check rules for configuration parameters and improve validity check of the configuration items
- Debugging interface
 - Add the `Drop Region` debugging interface
 - Add the interfaces to enumerate the health status of each PD
- Statistics
 - Add statistics about abnormal Regions
 - Add statistics about Region isolation level
 - Add scheduling related metrics
- Performance
 - Keep the PD leader and the etcd leader together in the same node to improve write performance
 - Optimize the performance of Region heartbeat

14.11.12.3 TiKV

- Features
 - Protect critical configuration from incorrect modification
 - Support `Region Merge experimental`
 - Add the `Raw DeleteRange` API
 - Add the `GetMetric` API
 - Add `Raw Batch Put`, `Raw Batch Get`, `Raw Batch Delete` and `Raw Batch Scan`
 - Add Column Family options for the RawKV API and support executing operation on a specific Column Family
 - Support Streaming and Streaming Aggregation in Coprocessor
 - Support configuring the request timeout of Coprocessor
 - Carry timestamps with Region heartbeats
 - Support modifying some RocksDB parameters online, such as `block-cache-size`
 - Support configuring the behavior of Coprocessor when it encounters some warnings or errors
 - Support starting in the importing data mode to reduce write amplification during the data importing process
 - Support manually splitting Region in halves
 - Improve the data recovery tool `tikv-ctl`

- Return more statistics in Coprocessor to guide the behavior of TiDB
- Support the `ImportSST` API to import SST files **experimental**
- Add the TiKV Importer binary to integrate with TiDB Lightning to import data quickly **experimental**
- Performance
 - Optimize read performance using `ReadPool` and increase the `raw_get/get/ ↳ batch_get` by 30%
 - Improve metrics performance
 - Inform PD immediately once the Raft snapshot process is completed to speed up balancing
 - Solve performance jitter caused by RocksDB flushing
 - Optimize the space reclaiming mechanism after deleting data
 - Speed up garbage cleaning while starting the server
 - Reduce the I/O overhead during replica migration using `DeleteFilesInRanges`
- Stability
 - Fix the issue that gRPC call does not get returned when the PD leader switches
 - Fix the issue that it is slow to offline nodes caused by snapshots
 - Limit the temporary space usage consumed by migrating replicas
 - Report the Regions that cannot elect a leader for a long time
 - Update the Region size information in time according to compaction events
 - Limit the size of scan lock to avoid request timeout
 - Limit the memory usage when receiving snapshots to avoid OOM
 - Increase the speed of CI test
 - Fix the OOM issue caused by too many snapshots
 - Configure `keepalive` of gRPC
 - Fix the OOM issue caused by an increase of the Region number

14.11.12.4 TiSpark

TiSpark uses a separate version number. The current TiSpark version is 1.0 GA. The components of TiSpark 1.0 provide distributed computing of TiDB data using Apache Spark.

- Provide a gRPC communication framework to read data from TiKV
- Provide encoding and decoding of TiKV component data and communication protocol
- Provide calculation pushdown, which includes:
 - Aggregate pushdown
 - Predicate pushdown
 - TopN pushdown
 - Limit pushdown
- Provide index related support
 - Transform predicate into Region key range or secondary index

- Optimize `Index Only` queries - Adaptively downgrade index scan to table scan per Region
- Provide cost-based optimization
 - Support statistics
 - Select index
 - Estimate broadcast table cost
- Provide support for multiple Spark interfaces
 - Support Spark Shell
 - Support ThriftServer/JDBC
 - Support Spark-SQL interaction
 - Support PySpark Shell
 - Support SparkR

14.11.13 TiDB 2.0 RC5 Release Notes

On April 17, 2018, TiDB 2.0 RC5 is released. This release has great improvement in MySQL compatibility, SQL optimization and stability.

14.11.13.1 TiDB

- Fix the issue about applying the `Top-N` pushdown rule
- Fix the estimation of the number of rows for the columns that contain `NULL` values
- Fix the zero value of the `Binary` type
- Fix the `BatchGet` issue within a transaction
- Clean up the written data while rolling back the `Add Index` operation, to reduce consumed space
- Optimize the `insert on duplicate key update` statement to improve the performance by 10 times
- Fix the issue about the type of the results returned by the `UNIX_TIMESTAMP` function
- Fix the issue that the `NULL` value is inserted while adding `NOT NULL` columns
- Support showing memory usage of the executing statements in the `Show Process List` statement
- Fix the issue that `Alter Table Modify Column` reports an error in extreme conditions
- Support setting the table comment using the `Alter` statement

14.11.13.2 PD

- Add support for Raft Learner
- Optimize the Balance Region Scheduler to reduce scheduling overhead
- Adjust the default value of `schedule-limit` configuration
- Fix the issue of allocating ID frequently
- Fix the compatibility issue when adding a new scheduler

14.11.13.3 TiKV

- Support the Region specified by `compact` in `tikv-ctl`
- Support Batch Put, Batch Get, Batch Delete and Batch Scan in the RawKVClient
- Fix the OOM issue caused by too many snapshots
- Return more detailed error information in Coprocessor
- Support dynamically modifying the `block-cache-size` in TiKV through `tikv-ctl`
- Further improve `importer`
- Simplify the `ImportSST::Upload` interface
- Configure the `keepalive` property of gRPC
- Split `tikv-importer` from TiKV as an independent binary
- Provide statistics about the number of rows scanned by each `scan range` in Coprocessor
- Fix the compilation issue on the macOS system
- Fix the issue of misusing a RocksDB metric
- Support the `overflow as warning` option in Coprocessor

14.11.14 TiDB 2.0 RC4 Release Notes

On March 30, 2018, TiDB 2.0 RC4 is released. This release has great improvement in MySQL compatibility, SQL optimization and stability.

14.11.14.1 TiDB

- Support `SHOW GRANTS FOR CURRENT_USER();`
- Fix the issue that the `Expression` in `UnionScan` is not cloned
- Support the `SET TRANSACTION` syntax
- Fix the potential goroutine leak issue in `copIterator`
- Fix the issue that `admin check table` misjudges the unique index including null
- Support displaying floating point numbers using scientific notation
- Fix the type inference issue during binary literal computing
- Fix the issue in parsing the `CREATE VIEW` statement
- Fix the panic issue when one statement contains both `ORDER BY` and `LIMIT 0`
- Improve the execution performance of `DecodeBytes`
- Optimize `LIMIT 0` to `TableDual`, to avoid building useless execution plans

14.11.14.2 PD

- Support splitting Region manually to handle the hot spot in a single Region
- Fix the issue that the label property is not displayed when `pdctl` runs `config show ↴ all`
- Optimize metrics and code structure

14.11.14.3 TiKV

- Limit the memory usage during receiving snapshots, to avoid OOM in extreme conditions
- Support configuring the behavior of Coprocessor when it encounters warnings
- Support importing the data pattern in TiKV
- Support splitting Region in the middle
- Increase the speed of CI test
- Use `crossbeam channel`
- Fix the issue that too many logs are output caused by leader missing when TiKV is isolated

14.11.15 TiDB 2.0 RC3 Release Notes

On March 23, 2018, TiDB 2.0 RC3 is released. This release has great improvement in MySQL compatibility, SQL optimization and stability.

14.11.15.1 TiDB

- Fix the wrong result issue of `MAX/MIN` in some scenarios
- Fix the issue that the result of `Sort Merge Join` does not show in order of Join Key in some scenarios
- Fix the error of comparison between `uint` and `int` in boundary conditions
- Optimize checks on length and precision of the floating point type, to improve compatibility with MySQL
- Improve the parsing error log of time type and add more error information
- Improve memory control and add statistics about `IndexLookupExecutor` memory
- Optimize the execution speed of `ADD INDEX` to greatly increase the speed in some scenarios
- Use the Stream Aggregation operator when the `GROUP BY` substatement is empty, to increase the speed
- Support closing the `Join Reorder` optimization in the optimizer using `STRAIGHT_JOIN`
- Output more detailed status information of DDL jobs in `ADMIN SHOW DDL JOBS`
- Support querying the original statements of currently running DDL jobs using `ADMIN → SHOW DDL JOB QUERIES`
- Support recovering the index data using `ADMIN RECOVER INDEX` for disaster recovery
- Attach a lower priority to the `ADD INDEX` operation to reduce the impact on online business
- Support aggregation functions with JSON type parameters, such as `SUM/AVG`
- Support modifying the `lower_case_table_names` system variable in the configuration file, to support the OGG data replication tool
- Improve compatibility with the Navicat management tool
- Support using implicit RowID in CRUD operations

14.11.15.2 PD

- Support Region Merge, to merge empty Regions or small Regions after deleting data
- Ignore the nodes that have a lot of pending peers during adding replicas, to improve the speed of restoring replicas or making nodes offline
- Fix the frequent scheduling issue caused by a large number of empty Regions
- Optimize the scheduling speed of leader balance in scenarios of unbalanced resources within different labels
- Add more statistics about abnormal Regions

14.11.15.3 TiKV

- Support Region Merge
- Inform PD immediately once the Raft snapshot process is completed, to speed up balancing
- Add the Raw DeleteRange API
- Add the GetMetric API
- Reduce the I/O fluctuation caused by RocksDB sync files
- Optimize the space reclaiming mechanism after deleting data
- Improve the data recovery tool `tikv-ctl`
- Fix the issue that it is slow to make nodes down caused by snapshot
- Support streaming in Coprocessor
- Support Readpool and increase the `raw_get/get/batch_get` by 30%
- Support configuring the request timeout of Coprocessor
- Support streaming aggregation in Coprocessor
- Carry time information in Region heartbeats
- Limit the space usage of snapshot files to avoid consuming too much disk space
- Record and report the Regions that cannot elect a leader for a long time
- Speed up garbage cleaning when starting the server
- Update the size information about the corresponding Region according to compaction events
- Limit the size of `scan lock` to avoid request timeout
- Use `DeleteRange` to speed up Region deletion
- Support modifying RocksDB parameters online

14.11.16 TiDB 2.0 RC1 Release Notes

On March 9, 2018, TiDB 2.0 RC1 is released. This release has great improvement in MySQL compatibility, SQL optimization and stability.

14.11.16.1 TiDB

- Support limiting the memory usage by a single SQL statement, to reduce the risk of OOM

- Support pushing the Stream Aggregate operator down to TiKV
- Support validating the configuration file
- Support obtaining the information of TiDB configuration through HTTP API
- Compatible with more MySQL syntax in Parser
- Improve the compatibility with Navicat
- Improve the optimizer and extract common expressions with multiple OR conditions, to choose better query plan
- Improve the optimizer and convert subqueries to Join operators in more scenarios, to choose better query plan
- Resolve Lock in the Batch mode to increase the garbage collection speed
- Fix the length of Boolean field to improve compatibility
- Optimize the Add Index operation and give lower priority to all write and read operations, to reduce the impact on online business

14.11.16.2 PD

- Optimize the logic of code used to check the Region status to improve performance
- Optimize the output of log information in abnormal conditions to facilitate debugging
- Fix the monitor statistics that the disk space of TiKV nodes is not enough
- Fix the wrong reporting issue of the health interface when TLS is enabled
- Fix the issue that concurrent addition of replicas might exceed the threshold value of configuration, to improve stability

14.11.16.3 TiKV

- Fix the issue that gRPC call is not cancelled when PD leaders switch
- Protect important configuration which cannot be changed after initial configuration
- Add gRPC APIs used to obtain metrics
- Check whether SSD is used when you start the cluster
- Optimize the read performance using ReadPool, and improve the performance by 30% in the `raw get` test
- Improve metrics and optimize the usage of metrics

14.11.17 TiDB 1.1 Beta Release Notes

On February 24, 2018, TiDB 1.1 Beta is released. This release has great improvement in MySQL compatibility, SQL optimization, stability, and performance.

14.11.17.1 TiDB

- Add more monitoring metrics and refine the log
- Compatible with more MySQL syntax
- Support displaying the table creating time in `information_schema`

- Optimize queries containing the `MaxOneRow` operator
- Configure the size of intermediate result sets generated by Join, to further reduce the memory used by Join
- Add the `tidb_config` session variable to output the current TiDB configuration
- Fix the panic issue in the `Union` and `Index Join` operators
- Fix the wrong result issue of the `Sort Merge Join` operator in some scenarios
- Fix the issue that the `Show Index` statement shows indexes that are in the process of adding
- Fix the failure of the `Drop Stats` statement
- Optimize the query performance of the SQL engine to improve the test result of the Sysbench Select/OLTP by 10%
- Improve the computing speed of subqueries in the optimizer using the new execution engine; compared with TiDB 1.0, TiDB 1.1 Beta has great improvement in tests like TPC-H and TPC-DS

14.11.17.2 PD

- Add the Drop Region debug interface
- Support setting priority of the PD leader
- Support configuring stores with a specific label not to schedule Raft leaders
- Add the interfaces to enumerate the health status of each PD
- Add more metrics
- Keep the PD leader and the etcd leader together as much as possible in the same node
- Improve the priority and speed of restoring data when TiKV goes down
- Enhance the validity check of the `data-dir` configuration item
- Optimize the performance of Region heartbeat
- Fix the issue that hot spot scheduling violates label constraint
- Fix other stability issues

14.11.17.3 TiKV

- Traverse locks using offset + limit to avoid potential GC problems
- Support resolving locks in batches to improve GC speed
- Support GC concurrency to improve GC speed
- Update the Region size using the RocksDB compaction listener for more accurate PD scheduling
- Delete the outdated data in batches using `DeleteFilesInRanges`, to make TiKV start faster
- Configure the Raft snapshot max size to avoid the retained files taking up too much space
- Support more recovery operations in `tikv-ctl`
- Optimize the ordered flow aggregation operation
- Improve metrics and fix bugs

14.11.18 TiDB 1.1 Alpha Release Notes

On January 19, 2018, TiDB 1.1 Alpha is released. This release has great improvement in MySQL compatibility, SQL optimization, stability, and performance.

14.11.18.1 TiDB

- SQL parser
 - Support more syntax
- SQL query optimizer
 - Use more compact structure to reduce statistics info memory usage
 - Speed up loading statistics info when starting tidb-server
 - Provide more accurate query cost evaluation
 - Use Count-Min Sketch to estimate the cost of queries using unique index more accurately
 - Support more complex conditions to make full use of index
- SQL executor
 - Refactor all executor operators using Chunk architecture, improve the execution performance of analytical statements and reduce memory usage
 - Optimize performance of the `INSERT IGNORE` statement
 - Push down more types and functions to TiKV
 - Support more `SQL_MODE`
 - Optimize the `Load Data` performance to increase the speed by 10 times
 - Optimize the `Use Database` performance
 - Support statistics on the memory usage of physical operators
- Server
 - Support the PROXY protocol

14.11.18.2 PD

- Add more APIs
- Support TLS
- Add more cases for scheduling Simulator
- Schedule to adapt to different Region sizes
- Fix some bugs about scheduling

14.11.18.3 TiKV

- Support Raft learner
- Optimize Raft Snapshot and reduce the I/O overhead
- Support TLS
- Optimize the RocksDB configuration to improve performance
- Optimize `count (*)` and query performance of unique index in Coprocessor
- Add more failpoints and stability test cases
- Solve the reconnection issue between PD and TiKV
- Enhance the features of the data recovery tool `tikv-ctl`
- Support splitting according to table in Region
- Support the `Delete Range` feature
- Support setting the I/O limit caused by snapshot
- Improve the flow control mechanism

14.12 v1.0

14.12.1 TiDB 1.0.8 Release Notes

On February 11, 2018, TiDB 1.0.8 is released with the following updates:

14.12.1.1 TiDB

- Fix issues in the `Outer Join` result in some scenarios
- Optimize the performance of the `InsertIntoIgnore` statement
- Fix the issue in the `ShardRowID` option
- Add limitation (Configurable, the default value is 5000) to the DML statements number within a transaction
- Fix an issue in the Table/Column aliases returned by the `Prepare` statement
- Fix an issue in updating statistics delta
- Fix a panic error in the `Drop Column` statement
- Fix an DML issue when running the `Add Column After` statement
- Improve the stability of the GC process by ignoring the regions with GC errors
- Run GC concurrently to accelerate the GC process
- Provide syntax support for the `CREATE INDEX` statement

14.12.1.2 PD

- Reduce the lock overhead of the region heartbeats
- Fix the issue that a hot region scheduler selects the wrong Leader

14.12.1.3 TiKV

- Use `DeleteFilesInRanges` to clear stale data and improve the TiKV starting speed
- Using `Decimal` in Coprocessor sum
- Sync the metadata of the received Snapshot compulsorily to ensure its safety

To upgrade from 1.0.7 to 1.0.8, follow the rolling upgrade order of PD -> TiKV -> TiDB.

14.12.2 TiDB 1.0.7 Release Notes

On January 22, 2018, TiDB 1.0.7 is released with the following updates:

14.12.2.1 TiDB

- Optimize the `FIELD_LIST` command
- Fix data race of the information schema
- Avoid adding read-only statements to history
- Add the `session` variable to control the log query
- Fix the resource leak issue in statistics
- Fix the goroutine leak issue
- Add schema info API for the http status server
- Fix an issue about `IndexJoin`
- Update the behavior when `RunWorker` is false in DDL
- Improve the stability of test results in statistics
- Support `PACK_KEYS` syntax for the `CREATE TABLE` statement
- Add `row_id` column for the null pushdown schema to optimize performance

14.12.2.2 PD

- Fix possible scheduling loss issue in abnormal conditions
- Fix the compatibility issue with proto3
- Add the log

14.12.2.3 TiKV

- Support `Table Scan`
- Support the remote mode in tikv-ctl
- Fix the format compatibility issue of tikv-ctl proto
- Fix the loss of scheduling command from PD
- Add timeout in Push metric

To upgrade from 1.0.6 to 1.0.7, follow the rolling upgrade order of PD -> TiKV -> TiDB.

14.12.3 TiDB 1.0.6 Release Notes

On January 08, 2018, TiDB 1.0.6 is released with the following updates:

14.12.3.1 TiDB

- Support the `Alter Table Auto_Increment` syntax
- Fix the bug in Cost Based computation and the `Null Json` issue in statistics
- Support the extension syntax to shard the implicit row ID to avoid write hot spot for a single table
- Fix a potential DDL issue
- Consider the timezone setting in the `curtime`, `sysdate` and `curdate` functions
- Support the `SEPARATOR` syntax in the `GROUP_CONCAT` function
- Fix the wrong return type issue of the `GROUP_CONCAT` function.

14.12.3.2 PD

- Fix store selection problem of hot-region scheduler

14.12.3.3 TiKV

None.

To upgrade from 1.0.5 to 1.0.6, follow the rolling upgrade order of PD -> TiKV -> TiDB.

14.12.4 TiDB 1.0.5 Release Notes

On December 26, 2017, TiDB 1.0.5 is released with the following updates:

14.12.4.1 TiDB

- Add the max value for the current `Auto_Increment` ID in the `Show Create Table` statement.
- Fix a potential goroutine leak.
- Support outputting slow queries into a separate file.
- Load the `TimeZone` variable from TiKV when creating a new session.
- Support the schema state check so that the `Show Create Table` and `Analyze` statements process the public table/index only.
- The `set transaction read only` should affect the `tx_read_only` variable.
- Clean up incremental statistic data when rolling back.
- Fix the issue of missing index length in the `Show Create Table` statement.

14.12.4.2 PD

- Fix the issue that the leaders stop balancing under some circumstances.
 - 869
 - 874
- Fix potential panic during bootstrapping.

14.12.4.3 TiKV

- Fix the issue that it is slow to get the CPU ID using the `get_cpuid` function.
- Support the `dynamic-level-bytes` parameter to improve the space collection situation.

To upgrade from 1.0.4 to 1.0.5, follow the rolling upgrade order of PD -> TiKV -> TiDB.

14.12.5 TiDB 1.0.4 Release Notes

On December 11, 2017, TiDB 1.0.4 is released with the following updates:

14.12.5.1 TiDB

- Speed up the loading of the statistics when starting the `tidb-server`
- Improve the performance of the `show variables` statement
- Fix a potential issue when using the `Add Index` statement to handle the combined indexes
- Fix a potential issue when using the `Rename Table` statement to move a table to another database
- Accelerate the effectiveness for the `Alter/Drop User` statement

14.12.5.2 TiKV

- Fix a possible performance issue when a snapshot is applied
- Fix the performance issue for reverse scan after removing a lot of data
- Fix the wrong encoded result for the Decimal type under special circumstances

To upgrade from 1.0.3 to 1.0.4, follow the rolling upgrade order of PD -> TiKV -> TiDB.

14.12.6 TiDB 1.0.3 Release Notes

On November 28, 2017, TiDB 1.0.3 is released with the following updates:

14.12.6.1 TiDB

- Optimize the performance in transaction conflicts scenario
- Add the `TokenLimit` option in the config file
- Output the default database in slow query logs
- Remove the DDL statement from query duration metrics
- Optimize the query cost estimation
- Fix the index prefix issue when creating tables
- Support pushing down the expressions for the `Float` type to TiKV
- Fix the issue that it is slow to add index for tables with discrete integer primary index
- Reduce the unnecessary statistics updates
- Fix a potential issue during the transaction retry

14.12.6.2 PD

- Support adding more types of schedulers using API

14.12.6.3 TiKV

- Fix the deadlock issue with the PD client
- Fix the issue that the wrong leader value is prompted for `NotLeader`
- Fix the issue that the chunk size is too large in the coprocessor

To upgrade from 1.0.2 to 1.0.3, follow the rolling upgrade order of PD -> TiKV -> TiDB.

14.12.7 TiDB 1.0.2 Release Notes

On November 13, 2017, TiDB 1.0.2 is released with the following updates:

14.12.7.1 TiDB

- Optimize the cost estimation of index point query
- Support the `Alter Table Add Column (ColumnDef ColumnPosition)` syntax
- Optimize the queries whose `where` conditions are contradictory
- Optimize the `Add Index` operation to rectify the progress and reduce repetitive operations
- Optimize the `Index Look Join` operator to accelerate the query speed for small data size
- Fix the issue with prefix index judgment

14.12.7.2 Placement Driver (PD)

- Improve the stability of scheduling under exceptional situations

14.12.7.3 TiKV

- Support splitting table to ensure one region does not contain data from multiple tables
- Limit the length of a key to be no more than 4 KB
- More accurate read traffic statistics
- Implement deep protection on the coprocessor stack
- Fix the LIKE behavior and the `do_div_mod` bug

14.12.8 TiDB 1.0.1 Release Notes

On November 1, 2017, TiDB 1.0.1 is released with the following updates:

14.12.8.1 TiDB

- Support canceling DDL Job.
- Optimize the `IN` expression.
- Correct the result type of the `Show` statement.
- Support log slow query into a separate log file.
- Fix bugs.

14.12.8.2 TiKV

- Support flow control with write bytes.
- Reduce Raft allocation.
- Increase coprocessor stack size to 10MB.
- Remove the useless log from the coprocessor.

14.12.9 TiDB 1.0 Release Notes

On October 16, 2017, TiDB 1.0 is now released! This release is focused on MySQL compatibility, SQL optimization, stability, and performance.

14.12.9.1 TiDB

- The SQL query optimizer:
 - Adjust the cost model
 - Analyze pushdown
 - Function signature pushdown
- Optimize the internal data format to reduce the interim data size
- Enhance the MySQL compatibility
- Support the `NO_SQL_CACHE` syntax and limit the cache usage in the storage engine
- Refactor the Hash Aggregator operator to reduce the memory usage
- Support the Stream Aggregator operator

14.12.9.2 PD

- Support read flow based balancing
- Support setting the Store weight and weight based balancing

14.12.9.3 TiKV

- Coprocessor now supports more pushdown functions
- Support pushing down the sampling operation
- Support manually triggering data compact to collect space quickly
- Improve the performance and stability
- Add a Debug API for debugging
- TiSpark Beta Release:
- Support configuration framework
- Support ThriftServer/JDBC and Spark SQL

14.12.9.4 Acknowledgement

14.12.9.4.1 Special thanks to the following enterprises and teams

- Archon
- Mobike
- Samsung Electronics
- SpeedyCloud
- Tencent Cloud
- UCloud

14.12.9.4.2 Thanks to the open source software and services from the following organizations and individuals

- Asta Xie
- CNCF
- CoreOS
- Databricks
- Docker
- Github
- Grafana
- gRPC
- Jepsen
- Kubernetes
- Namazu
- Prometheus
- RedHat
- RocksDB Team
- Rust Team

14.12.9.4.3 Thanks to the individual contributors

- 8cbx
- Akihiro Suda
- aliyx
- alston11111
- andelf
- Andy Librian
- Arthur Yang
- astaxie
- Bai, Yang
- bailaohe
- Bin Liu
- Blame cosmos
- Breezewish
- Carlos Ferreira
- Ce Gao
- Changjian Zhang
- Cheng Lian
- Cholerae Hu
- Chu Chao
- coldwater
- Cole R Lawrence
- cuiqiu
- cuiyuan
- Cwen
- Dagang
- David Chen
- David Ding
- dawxy
- dcadevil
- Deshi Xiao
- Di Tang
- disksing
- dongxu
- dreamquster
- Drogon
- Du Chuan
- Dylan Wen
- eBoyy
- Eric Romano
- Ewan Chou
- Fiisio
- follitude
- Fred Wang

- fud
- fudali
- gaoyangxiaozhu
- Gogs
- goroutine
- Gregory Ian
- Guanqun Lu
- Guilherme Hübner Franco
- Haibin Xie
- Han Fei
- hawkingrei
- Hiroaki Nakamura
- hiwjd
- Hongyuan Wang
- Hu Ming
- Hu Ziming
- Huachao Huang
- HuaiyuXu
- Huxley Hu
- iamxy
- Ian
- insion
- iroi44
- Ivan.Yang
- Jack Yu
- jacky liu
- Jan Mercl
- Jason W
- Jay
- Jay Lee
- Jianfei Wang
- Jiaxing Liang
- Jie Zhou
- jinhelin
- Jonathan Boulle
- Karl Ostendorf
- knarfeh
- Kuiba
- leixuechun
- li
- Li Shihai
- Liao Qiang
- Light
- lijian
- Lilian Lee

- Liqueur Librazy
- Liu Cong
- Liu Shaohui
- liubo0127
- liyanan
- lkk2003rty
- Louis
- louishust
- luckcolors
- Lynn
- Mae Huang
- maiyang
- maxwell
- mengshangqi
- Michael Belenchenco
- mo2zie
- morefreeze
- MQ
- mxlxm
- Neil Shen
- netroby
- ngaut
- Nicole Nie
- nolouch
- onlymellb
- overvenus
- PaladinTyrion
- paulg
- Priya Seth
- qgxiaozhan
- qhsong
- Qiannan
- qiukeren
- qiuyesuifeng
- queenypingcap
- qupeng
- Rain Li
- ranxiaolong
- Ray
- Rick Yu
- shady
- ShawnLi
- Shen Li
- Sheng Tang
- Shirly

- Shuai Li
- ShuNing
- ShuYu Wang
- siddontang
- silenceper
- Simon J Mudd
- Simon Xia
- skimmilk6877
- slt
- soup
- Sphinx
- Steffen
- sumBug
- sunhao2017
- Tao Meng
- Tao Zhou
- tennix
- tiancaiamao
- TianGuangyu
- Tristan Su
- ueizhou
- UncP
- Unknwon
- v01dstar
- Van
- WangXiangUSTC
- wangyanjun
- wangyisong1996
- weekface
- wegel
- Wei Fu
- Wenbin Xiao
- Wenting Li
- Wenxuan Shi
- winkyao
- woodpenker
- wuxuelian
- Xiang Li
- xiaojian cai
- Xuanjia Yang
- Xuanwo
- XuHuaiyu
- Yang Zhexuan
- Yann Autissier
- Yanzhe Chen

- Yiding Cui
- Yim
- youyouhu
- Yu Jun
- Yuwen Shen
- Zejun Li
- Zhang Yuning
- zhangjinpeng1987
- ZHAO Yijun
- Zhe-xuan Yang
- ZhengQian
- ZhengQianFang
- zhengwanbo
- ZhiFeng Hu
- Zhiyuan Zheng
- Zhou Tao
- Zhoubirdblue
- zhouniningnan
- Ziyi Yan
- zs634134578
- zxylvp
- zyguan
- zz-jason

14.12.10 Pre-GA Release Notes

On August 30, 2017, TiDB Pre-GA is released! This release is focused on MySQL compatibility, SQL optimization, stability, and performance.

14.12.10.1 TiDB

- The SQL query optimizer:
 - Adjust the cost model
 - Use index scan to handle the `where` clause with the `compare` expression which has different types on each side
 - Support the Greedy algorithm based Join Reorder
- Many enhancements have been introduced to be more compatible with MySQL
- Support `Natural Join`
- Support the JSON type (Experimental), including the query, update and index of the JSON fields
- Prune the useless data to reduce the consumption of the executor memory
- Support configuring prioritization in the SQL statements and automatically set the prioritization for some of the statements according to the query type
- Completed the expression refactor and the speed is increased by about 30%

14.12.10.2 Placement Driver (PD)

- Support manually changing the leader of the PD cluster

14.12.10.3 TiKV

- Use dedicated Rocksdb instance to store Raft log
- Use `DeleteRange` to speed up the deleting of replicas
- Coprocessor now supports more pushdown operators
- Improve the performance and stability

14.12.10.4 TiDB Connector for Spark Beta Release

- Implement the predicates pushdown
- Implement the aggregation pushdown
- Implement range pruning
- Capable of running full set of TPC+H except for one query that needs view support

14.12.11 TiDB RC4 Release Notes

On August 4, 2017, TiDB RC4 is released! This release is focused on MySQL compatibility, SQL optimization, stability, and performance.

14.12.11.1 Highlight

- For performance, the write performance is improved significantly, and the computing task scheduling supports prioritizing to avoid the impact of OLAP on OLTP.
- The optimizer is revised for a more accurate query cost estimating and for an automatic choice of the `Join` physical operator based on the cost.
- Many enhancements have been introduced to be more compatible with MySQL.
- TiSpark is now released to better support the OLAP business scenarios. You can now use Spark to access the data in TiKV.

14.12.11.2 Detailed updates

14.12.11.2.1 TiDB

- The SQL query optimizer refactoring:
 - Better support for TopN queries
 - Support the automatic choice of the `Join` physical operator based on the cost

- Improved Projection Elimination
- The version check of schema is based on Table to avoid the impact of DDL on the ongoing transactions
- Support `BatchIndexJoin`
- Improve the `Explain` statement
- Improve the `Index Scan` performance
- Many enhancements have been introduced to be more compatible with MySQL
- Support the JSON type and operations
- Support the configuration of query prioritizing and isolation level

14.12.11.2.2 Placement Driver (PD)

- Support using PD to set the TiKV location labels
- Optimize the scheduler
 - PD is now supported to initialize the scheduling commands to TiKV.
 - Accelerate the response speed of the region heartbeat.
 - Optimize the `balance` algorithm
- Optimize data loading to speed up failover

14.12.11.2.3 TiKV

- Support the configuration of query prioritizing
- Support the RC isolation level
- Improve Jepsen test results and the stability
- Support Document Store
- Coprocessor now supports more pushdown functions
- Improve the performance and stability

14.12.11.2.4 TiSpark Beta Release

- Implement the prediction pushdown
- Implement the aggregation pushdown
- Implement range pruning
- Capable of running full set of TPC-H except one query that needs view support

14.12.12 TiDB RC3 Release Notes

On June 16, 2017, TiDB RC3 is released! This release is focused on MySQL compatibility, SQL optimization, stability, and performance.

14.12.12.1 Highlight

- The privilege management is refined to enable users to manage the data access privileges using the same way as in MySQL.
- DDL is accelerated.
- The load balancing policy and process are optimized for performance.
- TiDB Ansible is open sourced. By using TiDB-Ansible, you can deploy, upgrade, start and shutdown a TiDB cluster with one click.

14.12.12.2 Detailed updates

14.12.12.3 TiDB

- The following features are added or improved in the SQL query optimizer:
 - Support incremental statistics
 - Support the `Merge Sort Join` operator
 - Support the `Index Lookup Join` operator
 - Support the `Optimizer Hint Syntax`
 - Optimize the memory consumption of the `Scan`, `Join`, `Aggregation` operators
 - Optimize the Cost Based Optimizer (CBO) framework
 - Refactor `Expression`
- Support more complete privilege management
- DDL acceleration
- Support using HTTP API to get the data distribution information of tables
- Support using system variables to control the query concurrency
- Add more MySQL built-in functions
- Support using system variables to automatically split a big transaction into smaller ones to commit

14.12.12.4 Placement Driver (PD)

- Support gRPC
- Provide the Disaster Recovery Toolkit
- Use Garbage Collection to clear stale data automatically
- Support more efficient data balance
- Support hot Region scheduling to enable load balancing and speed up the data importing
- Performance
 - Accelerate getting Client TSO
 - Improve the efficiency of Region Heartbeat processing
- Improve the `pd-ctl` function

- Update the Replica configuration dynamically
- Get the Timestamp Oracle (TSO)
- Use ID to get the Region information

14.12.12.5 TiKV

- Support gRPC
- Support the Sorted String Table (SST) format snapshot to improve the load balancing speed of a cluster
- Support using the Heap Profile to uncover memory leaks
- Support Streaming SIMD Extensions (SSE) and speed up the CRC32 calculation
- Accelerate transferring leader for faster load balancing
- Use Batch Apply to reduce CPU usage and improve the write performance
- Support parallel Prewrite to improve the transaction write speed
- Optimize the scheduling of the coprocessor thread pool to reduce the impact of big queries on point get
- The new Loader supports data importing at the table level, as well as splitting a big table into smaller logical blocks to import concurrently to improve the data importing speed.

14.12.13 TiDB RC2 Release Notes

On March 1, 2017, TiDB RC2 is released! This release is focused on the compatibility with MySQL, SQL query optimizer, system stability and performance in this version. What's more, a new permission management mechanism is added and users can control data access in the same way as the MySQL privilege management system.

14.12.13.1 TiDB

- Query optimizer
 - Collect column/index statistics and use them in the query optimizer
 - Optimize the correlated subquery
 - Optimize the Cost Based Optimizer (CBO) framework
 - Eliminate aggregation using unique key information
 - Refactor expression evaluation framework
 - Convert Distinct to GroupBy
 - Support the topn operation push-down
- Support basic privilege management
- Add lots of MySQL built-in functions
- Improve the Alter Table statement and support the modification of table name, default value and comment
- Support the Create Table Like statement

- Support the Show Warnings statement
- Support the Rename Table statement
- Restrict the size of a single transaction to avoid the cluster blocking of large transactions
- Automatically split data in the process of Load Data
- Optimize the performance of the AddIndex and Delete statement
- Support “ANSI_QUOTES” sql_mode
- Improve the monitoring system
- Fix Bugs
- Solve the problem of memory leak

14.12.13.2 PD

- Support location aware replica scheduling
- Conduct fast scheduling based on the number of region
- pd-ctl support more features
 - Add or delete PD
 - Obtain Region information with Key
 - Add or delete scheduler and operator
 - Obtain cluster label information

14.12.13.3 TiKV

- Support Async Apply to improve the entire write performance
- Use prefix seek to improve the read performance of Write CF
- Use memory hint prefix to improve the insert performance of Raft CF
- Optimize the single read transaction performance
- Support more push-down expressions
- Improve the monitoring system
- Fix Bugs

14.12.14 TiDB RC1 Release Notes

On December 23, 2016, TiDB RC1 is released. See the following updates in this release:

14.12.14.1 TiKV

- The write speed has been improved.
- The disk space usage is reduced.
- Hundreds of TBs of data can be supported.
- The stability is improved and TiKV can support a cluster with 200 nodes.
- Supports the Raw KV API and the Golang client.

14.12.14.2 Placement Driver (PD)

- The scheduling strategy framework is optimized and now the strategy is more flexible and reasonable.
- The support for `label` is added to support Cross Data Center scheduling.
- PD Controller is provided to operate the PD cluster more easily.

14.12.14.3 TiDB

- The following features are added or improved in the SQL query optimizer:
 - Eager aggregation
 - More detailed `EXPLAIN` information
 - Parallelization of the `UNION` operator
 - Optimization of the subquery performance
 - Optimization of the conditional push-down
 - Optimization of the Cost Based Optimizer (CBO) framework
- The implementation of the time related data types are refactored to improve the compatibility with MySQL.
- More built-in functions in MySQL are supported.
- The speed of the `add index` statement is enhanced.
- The following statements are supported:
 - Use the `CHANGE COLUMN` statement to change the name of a column.
 - Use `MODIFY COLUMN` and `CHANGE COLUMN` of the `ALTER TABLE` statement for some of the column type transfer.

14.12.14.4 New tools

- `Loader` is added to be compatible with the `mydumper` data format in Percona and provides the following functions:
 - Multi-thread import
 - Retry if error occurs
 - Breakpoint resume
 - Targeted optimization for TiDB
- The tool for one-click deployment is added.