Distributed Transactional Database

HotDB Server-

[Management Port Command]

Function Manual

Version No.: V2.5.6

Shanghai Hotpu Networks Technology Co., Ltd.

Nov. 2020

Trademark statement

Cautions

The products, services or features you purchase shall be subject to the commercial contracts and terms of Shanghai Hotpu Networks Technology Co., Ltd. The final interpretation right of the products, services or features in this document shall belong to Shanghai Hotpu Networks Technology Co., Ltd.

Shanghai Hotpu Networks Technology Co., Ltd.
Address: 603-605, Block A, SIM Technology building, No. 633, Jinzhong Road, Changning District, Shanghai
Postcode: 200050
Website: www.hotdb.com
Service email: service_hotdb@hotdb.cm
Service Tel: 021-5218 0789

Contents

1. New commands in V2.5.6

- show backupmasterdelay [DNID] -show master/slave replication delay of specified data node[DNID]
- disable_election -disable election in cluster
- enable_election-enable election in cluster
- disable_non_query_command -only allow query command
- enable_non_query_command -allow non-query command
- offline_to_dr -execute offline and online is not allowed
- exchangeconfig -exchange configuration of IDC
- exchangememoryconfig -exchange configuration in memory
- online_dr_check -check IDC switching
- online_dr_process -show IDC switching process
- check @@datasource_config_new -check MySQL parameter configuration information
- reset @@dberrorcount -clear all the error messages of LogicDBs

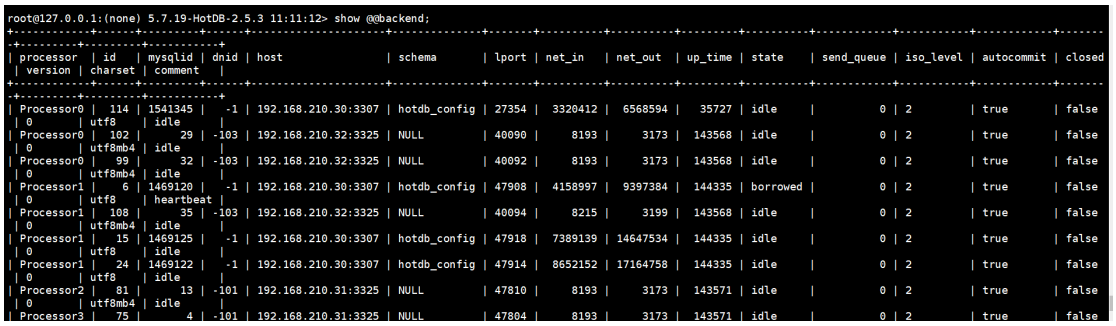
2. Data Detection Statement

2.1. HotDB Server statistics

2.1.1. show @@backend – show backend connection

This command is used to view the connection between HotDB Server and data source. For example:

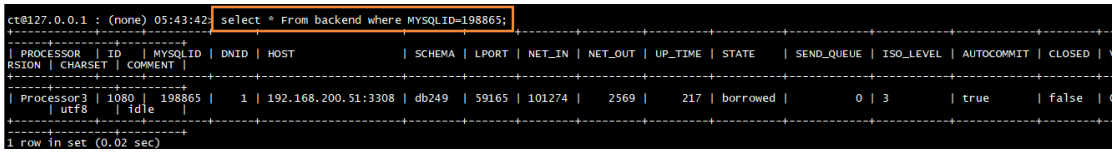
```
mysql> show @@backend;
```



```
root@127.0.0.1:(none) 5.7.19-HotDB-2.5.3 11:11:12> show @@backend;
+-----------+------+---------+------+--------------------+-------------+-------+---------+---------+---------+----------+-----------+----------+-----------+-------
-+---------+---------+-----------+
| processor | id   | mysqlid | dnid | host               | schema      | lport | net_in  | net_out | up_time | state    | send_queue | iso_level | autocommit | closed
 | version | charset | comment   |
+-----------+------+---------+------+--------------------+-------------+-------+---------+---------+---------+----------+-----------+----------+-----------+-------
-+---------+---------+-----------+
| Processor0 |  114 | 1541345 |   -1 | 192.168.210.30:3307 | hotdb_config | 27354 |  3320412 |  6568594 |   35727 | idle     |          0 | 2         | true       | false
 | 0       | utf8    | idle      |
| Processor0 |  102 |      29 | -103 | 192.168.210.32:3325 | NULL         | 40090 |     8193 |     3173 |  143568 | idle     |          0 | 2         | true       | false
 | 0       | utf8mb4 | idle      |
| Processor0 |   99 |      32 | -103 | 192.168.210.32:3325 | NULL         | 40092 |     8193 |     3173 |  143568 | idle     |          0 | 2         | true       | false
 | 0       | utf8mb4 | idle      |
| Processor1 |    6 | 1469120 |   -1 | 192.168.210.30:3307 | hotdb_config | 47908 |  4158997 |  9397384 |  144335 | borrowed |          0 | 2         | true       | false
 | 0       | utf8    | heartbeat |
| Processor1 |  108 |      35 | -103 | 192.168.210.32:3325 | NULL         | 40094 |     8215 |     3199 |  143568 | idle     |          0 | 2         | true       | false
 | 0       | utf8mb4 | idle      |
| Processor1 |   15 | 1469125 |   -1 | 192.168.210.30:3307 | hotdb_config | 47918 |  7389139 | 14647534 |  144335 | idle     |          0 | 2         | true       | false
 | 0       | utf8    | idle      |
| Processor1 |   24 | 1469122 |   -1 | 192.168.210.30:3307 | hotdb_config | 47914 |  8652152 | 17164758 |  144335 | idle     |          0 | 2         | true       | false
 | 0       | utf8    | idle      |
| Processor2 |   81 |      13 | -101 | 192.168.210.31:3325 | NULL         | 47810 |     8193 |     3173 |  143571 | idle     |          0 | 2         | true       | false
 | 0       | utf8mb4 | idle      |
| Processor3 |   75 |       4 | -101 | 192.168.210.31:3325 | NULL         | 47804 |     8193 |     3173 |  143571 | idle     |          0 | 2         | true       | false
```

Or query backend as you query a normal table:

```
mysql> select * from backend where MYSQLID=198865;
```



```
ct@127.0.0.1 : (none) 05:43:42> select * From backend where MYSQLID=198865;
+-----------+------+---------+------+--------------------+--------+-------+--------+---------+---------+----------+-----------+----------+-----------+--------+-
-----+---------+---------+
| PROCESSOR | ID   | MYSQLID | DNID | HOST               | SCHEMA | LPORT | NET_IN | NET_OUT | UP_TIME | STATE    | SEND_QUEUE | ISO_LEVEL | AUTOCOMMIT | CLOSED |
RSION | CHARSET | COMMENT |
+-----------+------+---------+------+--------------------+--------+-------+--------+---------+---------+----------+-----------+----------+-----------+--------+-
-----+---------+---------+
| Processor3 | 1080 |  198865 |    1 | 192.168.200.51:3308 | db249  | 59165 | 101274 |    2569 |     217 | borrowed |          0 | 3         | true       | false  |
     | utf8    | idle    |
+-----------+------+---------+------+--------------------+--------+-------+--------+---------+---------+----------+-----------+----------+-----------+--------+-
-----+---------+---------+
1 row in set (0.02 sec)
```

Or use HINT statement:

```
mysql> /*!hotdb:dnid=all*/select * from information_schema.processlist where info!='NULL' and id=198865;
```


```
1 row in set (0.01 sec)

ct@127.0.0.1 : TEST_CT 05:42:53> /*!hotdb:dnid=all*/select * from information_schema.processlist where info!='NULL' and id=198865;
+--------+---------------+-------------+------+---------+------+-----------+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| ID     | USER          | HOST        | DB   | COMMAND | TIME | STATE     | INFO                                                                                                                                                                  |
+--------+---------------+-------------+------+---------+------+-----------+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| 198865 | hotdb_datasource | ceph02:59165 | db249 | Query   |    0 | executing | select * from information_schema.processlist where info!='NULL' and id=198865 |
+--------+---------------+-------------+------+---------+------+-----------+-----------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.04 sec)

ct@127.0.0.1 : TEST_CT 05:43:55>
ct@127.0.0.1 : TEST_CT 05:51:47>
ct@127.0.0.1 : TEST_CT 05:51:47>
ct@127.0.0.1 : TEST_CT 05:51:47>
```

Fields and their description are contained in the result:

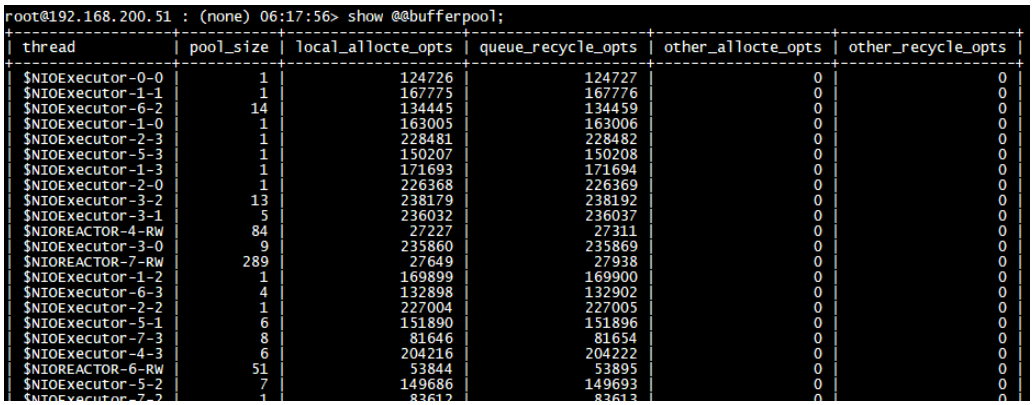
Column Name	Description	Value Type/Range
processor	The processor	STRING/[“Processor”number]
id	Backend connection id	LONG/[number]
mysqlid	Corresponding MySQL connection id	LONG/[number]
dnid	Data node id	INT/[number]
host	Host information	STRING/[host:port]
schema	Database name	STRING/[database]
lport	Local port	INT/[number]
net_in	Bytes received	LONG/[number]
net_out	Bytes sent	LONG/[number]
up_time	Uptime (s)	LONG/[number]
state	Connection status	connecting: the process of actively connecting to the server. A socket setup request is initiated, but not successful yet
		authenticating: handshake authentication process
		idle: idle available status
		borrowed: borrowed status: in the presence of a transaction scenario, even if the backend does not execute sql, the connection will still be held until commit and rollback are

		committed.
		running: a request is sent, and in the status of waiting for response or processing the response
		closed: connection is closed
send_queue	size of send queue	INT/[number]
iso_level	transaction isolation level	0: read uncommitted
		1: read committed
		2: repeatable read
		3: serializable
autocommit	autocommit or not	BOOLEAN/[true/false]
closed	closed or not	BOOLEAN/[true/false]
version	connection pool version number	INT/[number]
charset	result charset	STRING/[charset]
comment	comment	heartbeat: connection used by heartbeat
		latency check: connection used by latency detection
		idle: connection for idle status
		querying: connection for executing query

2.1.2. show @@bufferpool - Show the status of bufferpool

This command is used to view the status of bufferpool. For example:

```
mysql> show @@bufferpool;
```



```
root@192.168.200.51 : (none) 06:17:56> show @@bufferpool;
+-------------------+-----------+--------------------+--------------------+--------------------+--------------------+
| thread            | pool_size | local_allocte_opts | queue_recycle_opts | other_allocte_opts | other_recycle_opts |
+-------------------+-----------+--------------------+--------------------+--------------------+--------------------+
| $NIOExecutor-0-0  |         1 |             124726 |             124727 |                  0 |                  0 |
| $NIOExecutor-1-1  |         1 |             167775 |             167776 |                  0 |                  0 |
| $NIOExecutor-6-2  |        14 |             134445 |             134459 |                  0 |                  0 |
| $NIOExecutor-1-0  |         1 |             163005 |             163006 |                  0 |                  0 |
| $NIOExecutor-2-3  |         1 |             228481 |             228482 |                  0 |                  0 |
| $NIOExecutor-5-3  |         1 |             150207 |             150208 |                  0 |                  0 |
| $NIOExecutor-1-3  |         1 |             171693 |             171694 |                  0 |                  0 |
| $NIOExecutor-2-0  |         1 |             226368 |             226369 |                  0 |                  0 |
| $NIOExecutor-3-2  |        13 |             238179 |             238192 |                  0 |                  0 |
| $NIOExecutor-3-1  |         5 |             236032 |             236037 |                  0 |                  0 |
| $NIOREACTOR-4-RW  |        84 |              27227 |              27311 |                  0 |                  0 |
| $NIOExecutor-3-0  |         9 |             235860 |             235869 |                  0 |                  0 |
| $NIOREACTOR-7-RW  |       289 |              27649 |              27938 |                  0 |                  0 |
| $NIOExecutor-1-2  |         1 |             169899 |             169900 |                  0 |                  0 |
| $NIOExecutor-6-3  |         4 |             132898 |             132902 |                  0 |                  0 |
| $NIOExecutor-2-2  |         1 |             227004 |             227005 |                  0 |                  0 |
| $NIOExecutor-5-1  |         6 |             151890 |             151896 |                  0 |                  0 |
| $NIOExecutor-7-3  |         8 |              81646 |              81654 |                  0 |                  0 |
| $NIOExecutor-4-3  |         6 |             204216 |             204222 |                  0 |                  0 |
| $NIOREACTOR-6-RW  |        51 |              53844 |              53895 |                  0 |                  0 |
| $NIOExecutor-5-2  |         7 |             149686 |             149693 |                  0 |                  0 |
```

Fields and their description are contained in the result:

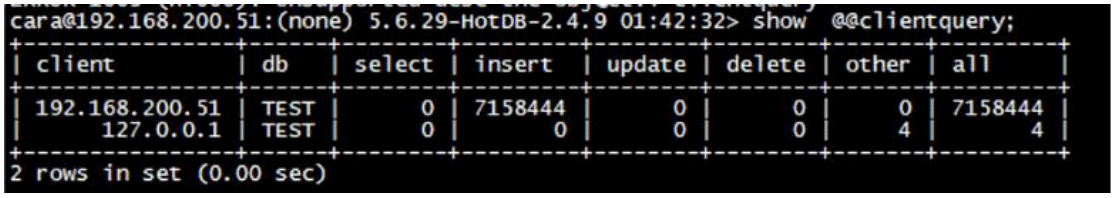
Column Name	Description	Value Type/Range
thread	thread name	STRING/ [“$NIOREACTOR-“[number]” -RW”, “$NIOExecutor-“[number]”-“ [number]]
pool_size	bufferpool size	INT/[number]
local_allocate_opts	The count of buffer requests of local cache thread	LONG /[number]
queue_recycle_opts	The count of buffer recycles of local cache thread	LONG/[number]
other_allocate_opts	The count of buffer requests of other threads	INT/[number]

other_recycle_opts	The count of buffer recycles of other threads	INT/[number]

2.1.3. show @@clientquery - statistics of current client query

This command is used to show the statistics of current client query. For example:

```
mysql> show @@clientquery;
```



Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
client	client information	STRING/[host]
db	LogicDB name	STRING/[database]
select	The count of query	LONG /[number]
insert	The count of insert	LONG /[number]
update	The count of update	LONG /[number]
delete	The count of delete	LONG /[number]
other	The count of other operations	LONG /[number]
all	all	LONG/[number]

Note: other counts the DDL statements executed by current client

2.1.4. show @@cluster – Show cluster member information

This command is used to view current cluster member status. This command is only used to view cluster member status, and has no reference value for single node and master/slave node. For example:

```
mysql> show @@cluster;
```

```
cara@127.0.0.1:(none) 5.6.29-HotDB-2.5.0 10:23:32> show @@cluster;
+-----------+----------------+------+-------------+--------------+
| status    | host           | port | server_port | manager_port |
+-----------+----------------+------+-------------+--------------+
| PRIMARY   | 192.168.210.23 | 3326 | 3323        | 3325         |
| SECONDARY | 192.168.210.24 | 3326 | 3323        | 3325         |
| SECONDARY | 192.168.210.22 | 3326 | 3323        | 3325         |
+-----------+----------------+------+-------------+--------------+
3 rows in set (0.00 sec)
```

Fields and their description are contained in the result:

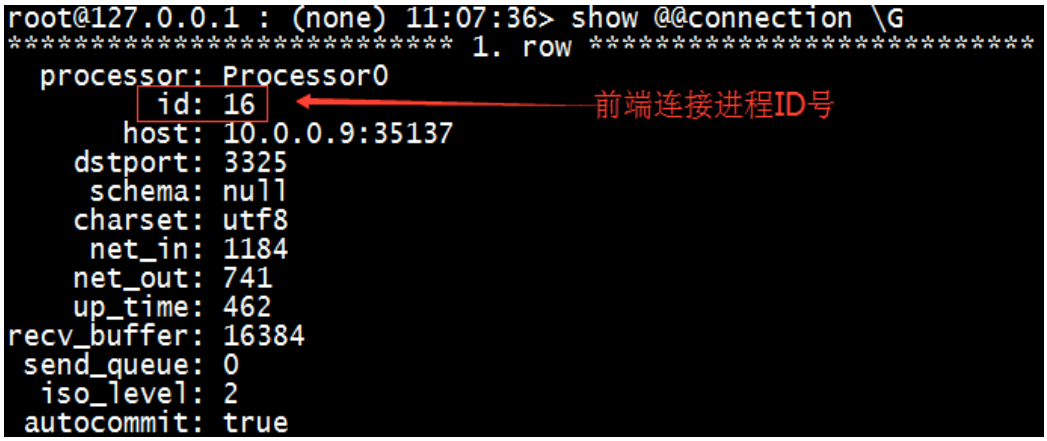
Column Name	Description	Value Type/Range
status	member status	STRING
host	member host	STRING/[host]
port	cluster communication port	INTEGER/[port]
server_port	cluster node server port	INTEGER/[port]
manager_port	cluster node Management Port	INTEGER/[port]

2.1.5. show @@connection – show frontend connection

This command is used to obtain the frontend connection of HotDB Server. For

example:

```
mysql> show @@connection;
```



Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
processor	processor name	STRING/[“Processor”number]
id	frontend connection id	LONG/[number]
host	client information	STRING/[host:port]
dstport	target port number	INT/[number]
schema	target database name	STRING/[database]
charset	charset	STRING/[charset]
net_in	bytes received	LONG/[number]
net_out	bytes sent	LONG/[number]
up_time	uptime (s)	INT/[number]
recv_buffer	size of receive queue (byte)	LONG/[number]
send_queue	size of send queue (byte)	LONG/[number]

iso_level	transaction isolation level	0: read uncommitted
		1: read committed
		2: repeatable read
		3: serializable
autocommit	autocommit or not	BOOLEAN/[true/false]

2.1.6. show @@connection_statistics – Show current live frontend connection statistics

This command is used to obtain current live frontend connection statistics of HotDB Server. For example:

```
mysql> show @@connection_statistics;
```

```
root@192.168.200.2 : (none) 03:37:21> show @@connection_statics;
+-------+-------------+-------+----------+----------+------+---------------------+---------------------+-----------------+
| id    | client_addr | port  | logicdb  | username | host | connect_time        | close_time          | operation_count |
+-------+-------------+-------+----------+----------+------+---------------------+---------------------+-----------------+
| 12771 | 127.0.0.1   | 31144 | TEST_ZTM | rmb      | %    | 2019-01-03 14:02:00.753 | 2019-01-03 15:37:28.084 | 8               |
| 12951 | 127.0.0.1   | 36222 | ZJJ_DB1  | jing01   | %    | 2019-01-03 15:11:59.005 | 2019-01-03 15:37:28.084 | 0               |
+-------+-------------+-------+----------+----------+------+---------------------+---------------------+-----------------+
2 rows in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
id	connection id	INTEGER/[number]
client_addr	client ip address	STRING/[host]
port	client connection port	INTEGER/[number]
logicdb	LogicDB used	STRING/[database]
username	username	STRING

host	host matched with client	STRING
connect_time	connection establishment time	STRING/[date]
close_time	current connection time	STRING/[date]
operation_count	The count of operations of this connection	INTEGER/[number]

2.1.7. show @@database – show current available LogicDB information

This command is used to show current available LogicDB information, which is equivalent to the command show databases under the MySQL. For example:

```
mysql> show @@database;
```

```
root@127.0.0.1 : (none) 03:24:17> show @@database;
+----------+
| database |
+----------+
| MYDB     |
+----------+
1 row in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
database	LogicDB	STRING/[database]

2.1.8. show @@datanode - show data node information

This command is used to show the node information of current database. For example:

```
mysql> show @@database;
```

```
root@127.0.0.1:(none) 5.7.19-HotDB-2.5.3 02:39:43> show @@datanode\G
*************************** 1. row ***************************
                    dn: -1
                  name: configDatanode
                    ds: 192.168.210.30:3307/hotdb_config
                 ds_id: -1
                  type: 1
                active: 3
                  idle: 5
                  size: 8
                 state: NORMAL
last_failover_start_time: NULL
  last_failover_duration: NULL
    last_failover_reason: NULL
      last_failover_info: NULL
            negotiation: OK
1 row in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
dn	data node number (the command restart @@heartbeat can be sued to restore heartbeat detection)	INT/[number]
ds	current data source information	STRING/[host:port/database]
ds_id	current data source id	INT/[number]
type	current data source type	1: Active Master

		2: Master/Slave
		3: Standby Slave
		4：MGR
active	active connections	INT/[number]
idle	idle connections	INT/[number]
size	all connections	INT/[number]
state	node status	normal: normal
		Failover: failover
last_failover_ start_time	start time of last failover	STRING/[yyyy-MM-dd HH:mm:ss.SSS]
last_failover_ duration	duration of last failover (ms)	STRING/[number]
last_failover_ reason	reason for last failover	STRING
last_failover_ info	information of last failover	STRING
negotiation	MGR node negotiation status	OK: normal
		ERROR: abnormal
		NULL: non-MGR

2.1.9. show @@datasource – show data source information

This command is used to view configuration information and status of current data source. For example:

```
mysql> show @@datasource;
```

```
root@127.0.0.1 : mydb 03:36:03> show @@datasource;
+------+------+------+--------+----------+------+--------+--------+------+------+
| dn   | ds   | type | status | host     | port | schema | active | idle | size |
+------+------+------+--------+----------+------+--------+--------+------+------+
| 1    | 1    | 1    | 1      | 10.0.0.9 | 3306 | db01   |      1 |   15 |   16 |
| 1    | 4    | 3    | 1      | 10.0.0.9 | 3307 | db01   |      1 |   15 |   16 |
| 2    | 2    | 1    | 1      | 10.0.0.9 | 3306 | db02   |      1 |   15 |   16 |
| 2    | 5    | 3    | 1      | 10.0.0.9 | 3307 | db02   |      1 |   15 |   16 |
| 3    | 3    | 1    | 1      | 10.0.0.9 | 3306 | db03   |      1 |   15 |   16 |
| 3    | 6    | 3    | 1      | 10.0.0.9 | 3307 | db03   |      1 |   15 |   16 |
+------+------+------+--------+----------+------+--------+--------+------+------+
6 rows in set (0.00 sec)
```

Fields and their description are contained in the result:

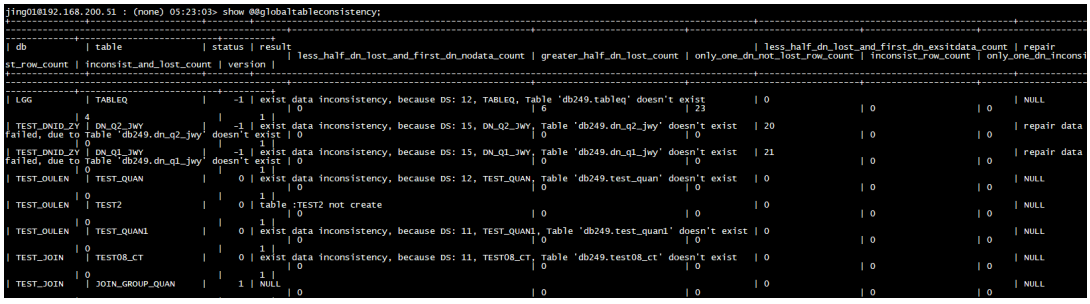
Column Name	Description	Value Type/Range
dn	data node number (the command restart @@heartbeat can be used to restore heartbeat detection)	INT/[number]
ds	current data source information	STRING/[host:port/database]
type	current data source type	1: Active Master
		2: Master/Slave
		3: Standby Slave
		4：MGR
status	data source status	0: unavailable
		1: available

		2: last data source abnormal
host	host address	STRING/[IP]
port	host port	STRING /[port]
schema	database name	STRING/[database]
active	active connections	INT/[number]
idle	idle connections	INT/[number]
size	all connections	INT/[number]
unavailable_reason	reasons for unavailable data source	STRING
flow_control	The count of remaining available	INT/[number]

2.1.10. show @@globaltableconsistency –global table consistency detection

This command is used to detect the consistency of global table. For example:

```
mysql> show @@globaltableconsistency;
```



Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
db	LogicDB name	STRING/[database]
table	global table name	STRING/[host:port/database]
status	status	0: undetectable
		1: consistency
		-1: inconsistency
result	detection result	STRING
less_half_dn_lost_and_first_dn_exsitdata_count	the number of lines with less than half nodes lost and with data on the first node	INT/[number]
repair	restoring state	STRING
less_half_dn_lost_and_first_dn_nodata_count	the number of lines with less than half nodes lost and without data on the first node	INT/[number]
greater_half_dn_lost_count	the number of lines with more than half nodes lost	INT/[number]
only_one_dn_not_lost_row_count	the number of lines with only one node not lost	INT/[number]
inconsist_row_count	the number of lines with more than one node inconsistent	INT/[number]

only_one_dn_inconsist_row _count	the number of lines with only one node inconsistent and none lost	INT/[number]
inconsist_and_lost_count	the number of lines with nodes inconsisntent and lost	INT/[number]
version	detection version	INT/[number]

2.1.11. show @@heartbeat – show backend heartbeat status

The command is used to report the heartbeat status. For example:

```
mysql> show @@heartbeat;
```

```
root@127.0.0.1 : (none) 11:44:14> show @@heartbeat\G
*************************** 1. row ***************************
            dn: 1
         ds_id: 1
       ds_type: master
          host: 10.0.0.9
          port: 3306
            db: db01
         retry: 0.0
        status: idle
        period: 2000
  execute_time: 0,0,35
last_active_time: 2017-05-26 11:50:28
          stop: false
*************************** 2. row ***************************
            dn: 1
         ds_id: 4
       ds_type: slave
          host: 10.0.0.9
          port: 3307
            db: db01
         retry: 0.0
        status: unknown
        period: 2000
  execute_time: 0,0,0
last_active_time: 1970-01-01 08:00:00
          stop: false
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
dn	data node id	INT/[number]
ds_id	data source id	INT/[number]
ds_type	data source type	STRING/[master/slave]
host	host address	STRING/[ip]
port	host port	INT/[port]
db	database name	STRING/[database]
retry	number of retries	INT/[number]
status	heartbeat status	checking: checking
		idle: heartbeat detection is normally started
		stopped: stopped
		paused: heartbeat detection is paused
		unknown: heartbeat detection function is not started
period	heartbeat period	INT/[number]
execute_time	average heartbeat response time of recent 10s, 1min and 5min (ms)	STRING/[number],[number],[number]
last_active_time	lastest heartbeat success time	DATETIME/[yyyy-MM-dd HH:mm:ss]
stop	heartbeat stops or not	BOOLEAN/[true/false]

Note: dn=-1 means configdb

2.1.12. show @@latency – show the synchronization latency

This command is used to view whether there is latency of master/slave database synchronization (the value can be shown only when the failover rule needs to be、configured). When there is latency of master/slave data, for example, when you set the SQL_DELAY time of standby slave:

```
           Master_SSL_Cipher:
              Master_SSL_Key:
       Seconds_Behind_Master: 60
Master_SSL_Verify_Server_Cert: No
               Last_IO_Errno: 0
               Last_IO_Error:
              Last_SQL_Errno: 0
              Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 2005133100
                  Master_UUID: 50216be0-9166-11e7-8c51-0026b961fdca
             Master_Info_File: /data/mysqldata3310/mydata/master.info
                    SQL_Delay: 60
          SQL_Remaining_Delay: 0
      Slave_SQL_Running_State: Waiting until MASTER_DELAY seconds after master executed event
           Master_Retry_Count: 86400
                  Master_Bind:
      Last_IO_Error_Timestamp:
     Last_SQL_Error_Timestamp:
               Master_SSL_Crl:
```

```
mysql> show @@latency;
```

```
root@192.168.200.51 : (none) 12:00:44> show @@latency;
+------+---------------------------+---------------------------+----------+
| dn   | info                      | backup_info               | latency  |
+------+---------------------------+---------------------------+----------+
| 3    | 192.168.200.51:3310/db249 | 192.168.200.52:3310/db249 | 59805 ms |
| 4    | 192.168.200.51:3311/db249 | 192.168.200.52:3311/db249 | 0 ms     |
| 21   | 192.168.200.52:3312/db249 | 192.168.200.51:3312/db249 | 0 ms     |
| 22   | 192.168.200.51:3313/db249 | 192.168.200.52:3313/db249 | 0 ms     |
+------+---------------------------+---------------------------+----------+
4 rows in set (0.03 sec)
```

If there is no latency, it shows:

```
root@127.0.0.1:(none) 5.1.27-HotDB-2.4.4 01:37:16> show @@latency;
+------+---------------------------+---------------------------+---------+
| dn   | info                      | backup_info               | latency |
+------+---------------------------+---------------------------+---------+
| 1    | 192.168.200.202:3306/db01 | 192.168.200.203:3310/db01 | 0 ms    |
| 2    | 192.168.200.202:3307/db01 | 192.168.200.203:3311/db01 | 0 ms    |
| 3    | 192.168.200.202:3308/db01 | 192.168.200.203:3312/db01 | 0 ms    |
| 4    | 192.168.200.202:3309/db01 | 192.168.200.203:3313/db01 | 0 ms    |
| 5    | 192.168.200.203:3306/db01 | 192.168.200.202:3310/db01 | 0 ms    |
| 6    | 192.168.200.203:3307/db01 | 192.168.200.202:3311/db01 | 0 ms    |
| 7    | 192.168.200.203:3308/db01 | 192.168.200.202:3312/db01 | 0 ms    |
| 8    | 192.168.200.203:3309/db01 | 192.168.200.202:3313/db01 | 0 ms    |
+------+---------------------------+---------------------------+---------+
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
dn	data node id	INT/[number]
info	current data source path	STRING/[ip]:[port]/[database]
backup_info	slave data source path	STRING/[ip]:[port]/[database]
latency	If it is available, it shows synchronization latency (ms); if it is unavailable or the data source is stopped, it shows “STOPPED”; if there is not synchronization latency, it shows “ERROR! Check your replication.”; if the synchronization latency is invalid, it shows “ERROR! Check your replication.(datasource may have just switched)”	STRING/[number] ms,”STOPPED”, “ERROR! Check your replication.”, “ERROR! Check your replication.(datasource may have just switched)”

2.1.13. show @@longtransaction – show long transaction information

This command is used to view the information of long transaction. For example:

```
mysql> show @@longtransaction;
```

```
ct@127.0.0.1 : (none) 03:13:33> show @@longtransaction;
+----------------+------+-------------+---------------------+
| host           | port | trx_id      | trx_started         |
+----------------+------+-------------+---------------------+
| 192.168.200.51 | 3309 |  4440168284 | 2018-06-05 15:13:23 |
| 192.168.200.51 | 3308 | 17273010040 | 2018-06-05 15:13:23 |
+----------------+------+-------------+---------------------+
2 rows in set (0.00 sec)
```

The long transaction is judged based on: transactions executed for more than 10s are all long transactions. Refer to the rules:

```
select    trx_id, trx_started from information_schema.innodb_trx where trx_started<=date_sub(now(),interval 10 second);
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
host	host address	STRING/[IP]
port	host port	INT/[PORT]
trx_id	transaction id	STRING/[number]

2.1.14. show @@masterslaveconsistency – master/slave data consistency detection

This command is used to show the consistency of tables in active master and standby slave. For example:

```
mysql> show @@masterslaveconsistency;
```

```
jing01@192.168.200.51 : (none) 05:36:58> show @@masterslaveconsistency;
+--------+-------------+-------+--------+-----------------------------------------------------------------------------------------------------------------------------+
| db     | table       | dn    | result | info                                                                                                                        |
+--------+-------------+-------+--------+-----------------------------------------------------------------------------------------------------------------------------+
| DDATA01 | TESTMASTER1 | dn_06 | NO     | Table :TESTMASTER1 in datanode: 15 exist data inconsistency where ID in range: 2-2;1-1; and inconsistent rows' primary key (ID) :(2),(1) |
| DDATA01 | PEOPLE      |       | YES    |                                                                                                                             |
| DDATA01 | ZJJ_TABLE02 |       | YES    |                                                                                                                             |
| DDATA01 | ZJJ_TABLE03 |       | YES    |                                                                                                                             |
| DDATA01 | PEOPLE2     |       | YES    |                                                                                                                             |
| DDATA01 | TESTMASTER  |       | YES    |                                                                                                                             |
| DDATA01 | PEOPLE1     |       | YES    |                                                                                                                             |
+--------+-------------+-------+--------+-----------------------------------------------------------------------------------------------------------------------------+
```

The above result shows that the master/salve data of DN_06 node is inconsistent.

```
cara@127.0.0.1 : (none) 05:27:21>  show @@masterslaveconsistency;
+---------------+------------------------------------+------+--------+--------------------------------------------------------------------
| db            | table                              | dn   | result | info
+---------------+------------------------------------+------+--------+--------------------------------------------------------------------
| LGG           | CC                                 | dn03 | UNKOWN | DN: 3 not exsit index of table: CC
| LGG           | CC                                 | dn04 | UNKOWN | DN: 4 not exsit index of table: CC
| TEST_JOIN     | DML_A_JWY                          | dn03 | NO     | DN: 3,Table structure is inconsistent, please check the table definition
```

and:

The above result shows that the CC table in the LogicDB LGG has no index defined, and consistency cannot be detected; DML_A_JWY table structure is inconsistent.

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
db	LogicDB name	STRING/[database]
table	table name	STRING/[table]
dn	data node name	STRING
result	consistent or not	STRING/[“YES”,“NO” ,“UNKNOWN”]
info	consistency result	STRING

2.1.15. show @@operation – show detailed command execution statistics

This command is used to show the data source actually used, and statistics of backend command execution. For example, frontend insert execution:

```
mysql> insert into tid values(10),(2);
```

```
mysql> insert into tid values(1677870),(233333333);
```

```
root@192.168.200.51 : test_ct 03:06:46> insert into tid values(10),(2);
Query OK, 2 rows affected (0.02 sec)
Records: 2  Duplicates: 0  Warnings: 0

root@192.168.200.51 : test_ct 03:07:16> insert into tid values(1677870),(233333333);
Query OK, 2 rows affected (0.01 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

The query result shows the current execution of backend for data source table:

```
mysql> select * from operation where `TABLE` like '%tid%';
```

```
root@192.168.200.51 : (none) 03:08:50> select * from operation where `TABLE` like '%tid%';
+---------+------+------+----------------+------+------+-------+--------+--------+--------+---------+--------+------+
| SCHEMA  | DN   | DS   | HOST           | PORT | DB   | TABLE | SELECT | INSERT | UPDATE | DELETE  | REPLACE| OTHER| ALL  |
+---------+------+------+----------------+------+------+-------+--------+--------+--------+---------+--------+------+
| TEST_CT | 1    | 11   | 192.168.200.51 | 3308 | db249| TID   |      3 |      1 |      0 |       0 |      0 |    9 |   13 |
| TEST_CT | 2    | 12   | 192.168.200.51 | 3309 | db249| TID   |      3 |      2 |      0 |       0 |      0 |    1 |    6 |
| TEST_CT | 3    | 13   | 192.168.200.51 | 3310 | db249| TID   |      0 |      1 |      0 |       0 |      0 |    0 |    1 |
| TEST_CT | 3    | 14   | 192.168.200.52 | 3310 | db249| TID   |      3 |      0 |      0 |       0 |      0 |    0 |    3 |
+---------+------+------+----------------+------+------+-------+--------+--------+--------+---------+--------+------+
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
schema	LogicDB name	STRING/[database]
dn	data node id	INT/[number]
ds	data source id	INT/[number]
host	data source host ip	STRING/[IP]
port	data source port	INT/[number]
db	database	STRING/[database]
table	table name	STRING/[table]
select	The count of SELECT the [table]	LONG/[number]
insert	The count of INSERT the [table]	LONG /[number]
update	The count of INSERT the [table]	LONG /[number]

delete	The count of DELETE the [table]	LONG /[number]
replace	The count of REPLACE the [table]	LONG /[number]
other	The count of other operations for [table] (The count of executing DDL statements)	LONG /[number]
all	Statistics of the above operations	LONG /[number]

2.1.16. show @@operation_db – show command execution with LogicDB as unit

This command is used to show statistics of command execution with LogicDB as unit.

For example:

```
mysql> show @@operation_db;
```

```
root@192.168.200.51 : (none) 03:30:31> show @@operation_db;
+--------------+--------+--------+--------+--------+---------+-------+------+
| db           | select | insert | update | delete | replace | other | all  |
+--------------+--------+--------+--------+--------+---------+-------+------+
| LGG          |    312 |      0 |      0 |      0 |       0 |   160 |  472 |
| TEST_DNID_ZY |    196 |      0 |      0 |      0 |       0 |    70 |  266 |
| TEST_OULEN   |    270 |      0 |      0 |      0 |       0 |   120 |  390 |
| TEST_ZY      |    120 |      0 |      0 |      0 |       0 |   110 |  230 |
| TEST_JOIN    |   1360 |      0 |      0 |      0 |       0 |   500 | 1860 |
| MHM_TEST     |      0 |      0 |      0 |      0 |       0 |    40 |   40 |
| TEST_DML_JWY |     90 |      0 |      0 |      0 |       0 |    60 |  150 |
| TEST_JZL     |    147 |     10 |      4 |      0 |       0 |   175 |  336 |
| TEST_CT      |    367 |      4 |      0 |      0 |       0 |   285 |  656 |
| LWG          |     18 |      0 |      0 |      0 |       0 |    40 |   58 |
| TEST_PERF    |    374 |      0 |      0 |      0 |       0 |    90 |  464 |
| TEST_ZYTS    |      0 |      0 |      0 |      0 |       0 |    10 |   10 |
```

Fields and their description are contained in the result:

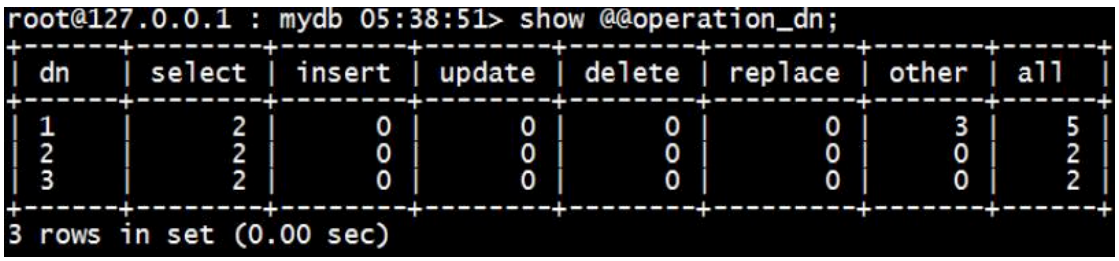
Column Name	Description	Value Type/Range
db	LogicDB name	STRING/[database]
select	The count of SELECT the [table]	LONG /[number]
insert	The count of INSERT the [table]	LONG /[number]
update	The count of INSERT the [table]	LONG /[number]
delete	The count of DELETE the [table]	LONG /[number]
replace	The count of REPLACE the [table]	LONG /[number]
other	The count of other operations for the table [table] (The count of executing DDL statements)	LONG /[number]
all	Statistics of the above operations	LONG /[number]

2.1.17. show @@operation_dn – show the command execution with data node as unit

This command is used to show the command execution statistis with data node as unit.

For example:

```
mysql> show @@operation_dn;
```



Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
dn	database node id	INT/[number]
select	The count of SELECT the [table]	LONG/[number]
insert	The count of INSERT the [table]	LONG/[number]
update	The count of INSERT the [table]	LONG/[number]
delete	The count of DELETE the [table]	LONG /[number]
replace	The count of REPLACE the [table]	LONG /[number]
other	The count of other operations for the table [table] (The count of executing DDL statement)	LONG /[number]
all	Statistics of the above operations	LONG/[number]

Note: the operations related to global table are separately counted according to operation types: only one node is counted for SELECT, all nodes are counted for INSERT, UPDATE and DELETE

2.1.18. show @@operation_ds – show command execution with data source as unit

This command is used to show the statistics of command execution with data source as unit. For example:

```
mysql> show @@operation_ds;
```

```
root@127.0.0.1 : mydb 05:41:54> show @@operation_ds;
+------+----------+------+------+--------+--------+--------+--------+---------+-------+------+
| ds   | host     | port | db   | select | insert | update | delete | replace | other | all  |
+------+----------+------+------+--------+--------+--------+--------+---------+-------+------+
| 1    | 10.0.0.9 | 3306 | db01 |      2 |      0 |      0 |      0 |       0 |     3 |    5 |
| 4    | 10.0.0.9 | 3307 | db01 |      0 |      0 |      0 |      0 |       0 |     0 |    0 |
| 2    | 10.0.0.9 | 3306 | db02 |      2 |      0 |      0 |      0 |       0 |     0 |    2 |
| 5    | 10.0.0.9 | 3307 | db02 |      0 |      0 |      0 |      0 |       0 |     0 |    0 |
| 3    | 10.0.0.9 | 3306 | db03 |      2 |      0 |      0 |      0 |       0 |     0 |    2 |
| 6    | 10.0.0.9 | 3307 | db03 |      0 |      0 |      0 |      0 |       0 |     0 |    0 |
+------+----------+------+------+--------+--------+--------+--------+---------+-------+------+
6 rows in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	/ Value Type/Range
ds	data source id	INT/[number]
host	data source host ip	STRING/[IP]
port	data source port	INT/[number]
db	database	STRING/[database]
select	The count of SELECT the [table]	LONG /[number]
insert	The count of INSERT the [table]	LONG /[number]
update	The count of INSERT the [table]	LONG /[number]
delete	The count of DELETE the [table]	LONG /[number]
replace	The count of REPLACE the [table]	LONG /[number]
other	The count of other operations for the table [table] (The count of executing DDL statements)	LONG /[number]
all	Statistics of the above operations	LONG /[number]

2.1.19. show @@operation_table – show the command execution with table as unit

This command is used to show the command execution statistics with LogicDB as unit. For example:

```
mysql> show @@operation_table;
```

```
root@192.168.200.51 : (none) 03:34:29> show @@operation_table;
+----------------------+--------+--------+--------+--------+---------+-------+-------+
| table                | select | insert | update | delete | replace | other | all   |
+----------------------+--------+--------+--------+--------+---------+-------+-------+
| TESTC                |     20 |      0 |      0 |      0 |       0 |    10 |    30 |
| TESTA                |     20 |      0 |      0 |      0 |       0 |    10 |    30 |
| TABLEA               |     50 |      0 |      0 |      0 |       0 |    10 |    60 |
| TABLEQ               |     80 |      0 |      0 |      0 |       0 |    10 |    90 |
| SPLIT                |     40 |      0 |      0 |      0 |       0 |    10 |    50 |
| MASTERSLAVE          |      0 |      0 |      0 |      0 |       0 |    10 |    10 |
| LGG_AUTOMOD          |      0 |      0 |      0 |      0 |       0 |    10 |    10 |
| SKIPDATATYPECHECK    |      0 |      0 |      0 |      0 |       0 |    10 |    10 |
| ABB                  |      0 |      0 |      0 |      0 |       0 |    10 |    10 |
| CBD                  |      0 |      0 |      0 |      0 |       0 |    10 |    10 |
| TEST_LWG_TABLE       |      0 |      0 |      0 |      0 |       0 |    10 |    10 |
| JOIN_DN01            |      0 |      0 |      0 |      0 |       0 |    10 |    10 |
| AA                   |      2 |      0 |      0 |      0 |       0 |    10 |    12 |
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
table	table name	STRING/[table]
select	The count of SELECT the [table]	LONG /[number]
insert	The count of INSERT the [table]	LONG /[number]
update	The count of INSERT the [table]	LONG /[number]
delete	The count of DELETE the [table]	LONG /[number]
replace	The count of REPLACE the [table]	LONG /[number]
other	The count of other operations for [table] (The count of executing	LONG /[number]

	DDL statements)	
all	Statistics of the above operations	LONG /[number]

2.1.20. show @@processor– show thread processing information

This command is used to view the thread processing information. For example:

```
mysql> show @@processor;
```

```
root@192.168.200.51 : (none) 10:53:09> show @@processor;
+------------+--------------+---------------+---------------+----------------+-----------+----------+---------+
| name       | front_net_in | front_net_out | backend_net_in | backend_net_out | frontends | backends | w_queue |
+------------+--------------+---------------+---------------+----------------+-----------+----------+---------+
| Processor0 |      7889034 |    1556807298 |    1137339408 |     1162183359 |         1 |       26 |       0 |
| Processor1 |     16245540 |    2757855217 |    1363850848 |     1057946566 |         5 |       14 |       0 |
| Processor2 |     13204200 |    4284698092 |    1413475642 |     1240559257 |         4 |       10 |       0 |
| Processor3 |     12205169 |    4358057023 |    1227211385 |     1171430612 |         2 |       17 |       0 |
| Processor4 |     10662276 |    2249175433 |    1256835587 |     1274330315 |         2 |       25 |       0 |
| Processor5 |      8648705 |    2032584240 |    1231054067 |     1282317920 |         1 |       22 |       0 |
| Processor6 |      7623412 |    1544737926 |    1395579486 |     1223052295 |         1 |       21 |       0 |
| Processor7 |      6680170 |    1888387594 |    1271471836 |     1452317616 |         1 |       27 |       0 |
+------------+--------------+---------------+---------------+----------------+-----------+----------+---------+
8 rows in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
name	processor name	STRING/[Processornumber]
front_net_in	fronend received bytes	LONG/[number]
front_net_out	frontend sent bytes	LONG/[number]
backend_net_in	backend received bytes	LONG/[number]
backend_net_out	backend sent bytes	LONG/[number]
frontends	frontend connections	LONG /[number]
backends	backend connections	LONG /[number]

w_queue	write queue size	LONG /[number]

2.1.21. show @@query – show frontend query statistics

This command is used to show the statistics of frontend command (excluding the Management Port). For example:

```
mysql> show @@query;
```

```
root@127.0.0.1 : mydb 05:50:37> show @@query;
+--------+--------+--------+--------+-------+------+
| select | insert | update | delete | other | all  |
+--------+--------+--------+--------+-------+------+
|      2 |      1 |      0 |      0 |     0 |    3 |
+--------+--------+--------+--------+-------+------+
1 row in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
select	The count of calling SELECT of this service	LONG /[number]
insert	The count of calling INSERT of this service	LONG /[number]
update	The count of calling UPDATE of this service	LONG /[number]
delete	The count of calling DELETE of this service	LONG /[number]
other	The count of calling other operations of this service (The count of executing DDL statement)	LONG /[number]

all	Statistics of the above operations	LONG /[number]

2.1.22. show @@query_db – show LogicDB frontend query statistics

This command is used to show statistics of command execution of each LogicDB. For example:

```
mysql> show @@query_db;
```

```
root@127.0.0.1 : mydb 05:51:49> show @@query_db;
+--------+--------+--------+--------+--------+-------+------+
| schema | select | insert | update | delete | other | all  |
+--------+--------+--------+--------+--------+-------+------+
| MYDB   |      2 |      1 |      0 |      0 |     0 |    3 |
+--------+--------+--------+--------+--------+-------+------+
1 row in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
schema	LogicDB	STRING/[database]
select	The count of SELECT the LogicDB [schema]	LONG /[number]
insert	The count of INSERT the LogicDB [schema]	LONG /[number]
update	The count of UPDATE the LogicDB [schema]	LONG /[number]

delete	The count of DELETE the LogicDB [schema]	LONG /[number]
other	The count of other operations for the LogicDB [schema] (The count of executing DDL statement)	LONG /[number]
all	Statistics of the above operations	LONG /[number]

2.1.23. show @@query_tb – show table-level frontend query statistics

This command is used to show statistics of command execution of each data table. For example:

```
mysql> show @@query_tb;
```

```
root@127.0.0.1 : mydb 05:52:50> show @@query_tb ;
+--------+-------+--------+--------+--------+--------+-------+------+
| schema | table | select | insert | update | delete | other | all  |
+--------+-------+--------+--------+--------+--------+-------+------+
| MYDB   | MYTB  |      2 |      1 |      0 |      0 |     0 |    3 |
+--------+-------+--------+--------+--------+--------+-------+------+
1 row in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
schema	LogicDB	STRING/[database]
table	table name	STRING/[table]
select	The count of SELECT the [table] under the	LONG /[number]

	LogicDB [schema]	
insert	The count of INSERT the [table] under the LogicDB [schema]	LONG /[number]
update	The count of UPDATE the [table] under the LogicDB [schema]	LONG /[number]
delete	The count of DELETE the [table] under the LogicDB [schema]	LONG /[number]
other	The count of other operations for the [table] under the LogicDB [schema] (The count of executing DDL statement)	LONG /[number]
all	Statistics of the above operations	LONG /[number]

2.1.24. show @@session – show current session information

This command is used to show current session information. For example:

```
mysql> show @@session;
```

```
root@127.0.0.1:(none) 5.7.19-HotDB-2.5.3 03:20:37> show @@session;
+------+---------+------------+---------+----------+----------+---------+---------+------+-----------+----------+----------+--------------+---------+---------+-----
--+----------+-------------------+--------------------+
| id   | running | trx_started | trx_time | trx_query | bk_count | bk_dnid | bk_dsid | bk_id | bk_mysqlid | bk_state | bk_closed | bk_autocommit | bk_host | bk_port | bk_d
b | bk_query | bk_last_read_time | bk_last_write_time |
+------+---------+------------+---------+----------+----------+---------+---------+------+-----------+----------+----------+--------------+---------+---------+-----
--+----------+-------------------+--------------------+
| 873  | FALSE   | NULL        | NULL     | NULL      | 0        | NULL    | NULL    | NULL  | NULL       | NULL     | NULL     | NULL         | NULL    | NULL    | NULL
  | NULL     | NULL              | NULL               |
| 876  | FALSE   | NULL        | NULL     | NULL      | 0        | NULL    | NULL    | NULL  | NULL       | NULL     | NULL     | NULL         | NULL    | NULL    | NULL
  | NULL     | NULL              | NULL               |
| 879  | FALSE   | NULL        | NULL     | NULL      | 0        | NULL    | NULL    | NULL  | NULL       | NULL     | NULL     | NULL         | NULL    | NULL    | NULL
  | NULL     | NULL              | NULL               |
| 897  | FALSE   | NULL        | NULL     | NULL      | 0        | NULL    | NULL    | NULL  | NULL       | NULL     | NULL     | NULL         | NULL    | NULL    | NULL
  | NULL     | NULL              | NULL               |
| 900  | FALSE   | NULL        | NULL     | NULL      | 0        | NULL    | NULL    | NULL  | NULL       | NULL     | NULL     | NULL         | NULL    | NULL    | NULL
  | NULL     | NULL              | NULL               |
| 882  | FALSE   | NULL        | NULL     | NULL      | 0        | NULL    | NULL    | NULL  | NULL       | NULL     | NULL     | NULL         | NULL    | NULL    | NULL
  | NULL     | NULL              | NULL               |
| 903  | FALSE   | NULL        | NULL     | NULL      | 0        | NULL    | NULL    | NULL  | NULL       | NULL     | NULL     | NULL         | NULL    | NULL    | NULL
  | NULL     | NULL              | NULL               |
| 906  | FALSE   | NULL        | NULL     | NULL      | 0        | NULL    | NULL    | NULL  | NULL       | NULL     | NULL     | NULL         | NULL    | NULL    | NULL
  | NULL     | NULL              | NULL               |
+------+---------+------------+---------+----------+----------+---------+---------+------+-----------+----------+----------+--------------+---------+---------+-----
--+----------+-------------------+--------------------+
8 rows in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
id	current session id	INT/[number]
running	SQL is in progress or not	BOOLEAN/[TRUE/FALSE]
trx_started	transaction start time	STRING/[yyyy-MM-dd HH:mm:ss.SSS]
trx_time	transaction duration (s)	INT/[number]
trx_query	last executed SQL	STRING/[SQL]
bk_count	total backend connections	INT/[number]
bk_dnid	backend connection node id	INT/[number]
bk_dsid	backend connection data source id	INT/[number]
bk_id	backend connection id	INT/[number]
bk_mysqlid	backend connection MySQL ID	INT/[number]
bk_state	backend connection status	STRING
bk_closed	backend connection is closed or not	BOOLEAN/[TRUE/FALSE]
bk_autocommit	backend connection is autocommitted or not	BOOLEAN/[TRUE/FALSE]
bk_host	backend connection Host	STRING/[host]
bk_port	backend connection port	INT/[port]
bk_db	backend connection database name	STRING/[DATABASE]
bk_query	the last executed SQL of	STRING/[SQL]

	backend connection	
bk_last_read_time	the last read time of backend connection	STRING/[yyyy-MM-dd HH:mm:ss.SSS]
bk_last_write_time	the last write time of backend connection	STRING/[yyyy-MM-dd HH:mm:ss.SSS]

2.1.25. show @@tableinfo – show table data information

This command is used to view the data information of each data table. For example:

```
mysql> show @@tableinfo;
```

```
root@127.0.0.1 : (none) 03:41:04> show @@tableinfo ;
+--------+------+------+----------+------+------+-------+------------+------------+-------------+
| schema | dn   | ds   | host     | port | db   | table | table_type | table_rows | data_length |
+--------+------+------+----------+------+------+-------+------------+------------+-------------+
| MYDB   | 1    | 1    | 10.0.0.9 | 3306 | db01 | mytb  | 1          |          5 |        8192 |
| MYDB   | 1    | 4    | 10.0.0.9 | 3307 | db01 | mytb  | 1          |          4 |       16384 |
| MYDB   | 2    | 2    | 10.0.0.9 | 3306 | db02 | mytb  | 1          |          6 |        8192 |
| MYDB   | 2    | 5    | 10.0.0.9 | 3307 | db02 | mytb  | 1          |          6 |       16384 |
| MYDB   | 3    | 3    | 10.0.0.9 | 3306 | db03 | mytb  | 1          |          2 |       16384 |
| MYDB   | 3    | 6    | 10.0.0.9 | 3307 | db03 | mytb  | 1          |          4 |       16384 |
+--------+------+------+----------+------+------+-------+------------+------------+-------------+
6 rows in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
schema	LogicDB	STRING/[database]
dn	data node id	INT/[number]
ds	data source id	INT/[number]
host	data source host ip	STRING/[IP]

port	data source port	INT/[PORT]
db	database	STRING/[database]
table	database name	STRING/[number]
table_type	table type	0: Global table
		1: Sharding table
table_rows	database rows	INT/[number]
data_length	data length (byte)	LONG/[number]

2.1.26. show @@tableinfo_db – show data information of table with LogicDB as unit

This command is used to view the data information of table with LogicDB as unit. For example:

```
mysql> show @@tableinfo_db;
```

```
root@127.0.0.1 : mydb 03:48:59> show @@tableinfo_db;
+------+------------+-------------+
| db   | table_rows | data_length |
+------+------------+-------------+
| MYDB |         27 |       81920 |
+------+------------+-------------+
1 row in set (0.01 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
db	LogicDB name	STRING/[database]

table_rows	database rows	INT/[number]
data_length	data length (byte)	LONG /[number]

2.1.27. show @@tableinfo_dn – show data information of table with data node as unit

This command is used to view the data information of table with data node as unit, and only the table information in current data source is counted. For example:

```
mysql> show @@tableinfo_dn
```

```
root@127.0.0.1 : mydb 03:50:05> show @@tableinfo_dn;
+------+------------+-------------+
| dn   | table_rows | data_length |
+------+------------+-------------+
| 3    |          6 |       32768 |
| 2    |         12 |       24576 |
| 1    |          9 |       24576 |
+------+------------+-------------+
3 rows in set (0.01 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
dn	data node id	INT/[number]
table_rows	database rows	INT/[number]
data_length	data length (byte)	LONG /[number]

2.1.28. show @@tableinfo_ds – show data information of table with data source as unit

This command is used to show the data information of table with data source as unit (including unavailable data source). For example:

```
mysql> show @@tableinfo_ds
```

```
root@127.0.0.1 : mydb 03:50:59> show @@tableinfo_ds;
+------+------------+-------------+
| ds   | table_rows | data_length |
+------+------------+-------------+
| 3    |          2 |       16384 |
| 2    |          6 |        8192 |
| 1    |          5 |        8192 |
| 6    |          4 |       16384 |
| 5    |          6 |       16384 |
| 4    |          4 |       16384 |
+------+------------+-------------+
6 rows in set (0.01 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
ds	data source id	INT/[number]
table_rows	database rows	INT/[number]
data_length	data length (byte)	LONG /[number]

2.1.29. show @@tableinfo_table – show table data information with table level

This command is used to show the data information of table with LogicDB as unit. For example:

```
mysql> show @@tableinfo_table;
```

```
root@127.0.0.1 : mydb 03:52:41> show @@tableinfo_table ;
+-------+------------+-------------+
| table | table_rows | data_length |
+-------+------------+-------------+
| mytb  |         27 |       81920 |
+-------+------------+-------------+
1 row in set (0.01 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
table	table name	STRING/[table]
table_rows	database rows	INT/[number]
data_length	data length (byte)	LONG /[number]

2.1.30. show @@threadpool – show status of threadpool

This command is used to view the status of threadpool. For example:

```
mysql> show @@threadpool;
```

```
root@192.168.200.51 : (none) 06:22:53> show @@threadpool;
+-----------------+-----------+--------------+-----------------+----------------+------------+
| name            | pool_size | active_count | task_queue_size | completed_task | total_task |
+-----------------+-----------+--------------+-----------------+----------------+------------+
| TimerExecutor   |         4 |            0 |              14 |       90436098 |   90436112 |
| $NIOExecutor-0- |         4 |            0 |               0 |         201487 |     201487 |
| $NIOExecutor-1- |         4 |            0 |               0 |         394082 |     394082 |
| $NIOExecutor-2- |         4 |            2 |               0 |         382408 |     382410 |
| $NIOExecutor-3- |         4 |            1 |               0 |         392079 |     392080 |
| $NIOExecutor-4- |         4 |            1 |               0 |         354179 |     354180 |
| $NIOExecutor-5- |         4 |            1 |               0 |         257486 |     257487 |
| $NIOExecutor-6- |         4 |            1 |               0 |         211886 |     211887 |
| $NIOExecutor-7- |         4 |            2 |               0 |         155974 |     155976 |
+-----------------+-----------+--------------+-----------------+----------------+------------+
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range

name	name of threadpool	STRING/"TimeExecutor","$NIOExecutor-" +number+"-"
pool_size	size of threadpool	INT/[number]
acive_count	The count of active thread	LONG/[number]
task_queue_size	size of task queue	LONG/[number]
completed_task	completed tasks	LONG/[number]
total_task	total tasks	LONG/[number]

2.1.31. show @@transaction – show transaction number

This command is used to view each LogicDB and count the number of currently completed autocomit and non-autocommit transactions, for example:

```
mysql> show @@transaction;
```

```
root@127.0.0.1:(none) 5.1.27-HotDB-2.4.4 01:29:39> show @@transaction;
+--------+-------------+
| schema | transaction |
+--------+-------------+
| HOTPU  | 30046647    |
+--------+-------------+
1 row in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
schema	LogicDB	STRING/[database]
transaction	the number of transactions	LONG/[number]

2.1.32. show hotdb datanodes – show current available nodes

This command is used to view the hotdb_datanodes table in configdb. The statement is:

```
mysql> show hotdb datanodes [LIKE 'pattern' | WHERE expr];
```

Parameter description:

Parameter	Description	Type
pattern	optional, fuzzy query expression, match the key rule_name	STRING
expr	optional, fuzzy query expression, match the specified key	STRING

For example:

```
zy@192.168.200.2:(none) 5.7.19-HotDB-2.5.1 02:51:13> show hotdb datanodes;
+-------------+---------------+---------------+
| datanode_id | datanode_name | datanode_type |
+-------------+---------------+---------------+
| 9           | dn_01         | 0             |
| 11          | dn_02         | 0             |
| 13          | dn_03         | 0             |
| 15          | dn_04         | 0             |
| 19          | dn_failover   | 0             |
| 20          | dn_rmb_01     | 0             |
+-------------+---------------+---------------+
6 rows in set (0.01 sec)
```

For another example:

```
mysql> show hotdb datanodes like 'dn_0%';
+-------------+---------------+---------------+
| datanode_id | datanode_name | datanode_type |
+-------------+---------------+---------------+
| 99          | dn_07         | 0             |
| 100         | dn_08         | 0             |
| 101         | dn_09         | 0             |
+-------------+---------------+---------------+
3 rows in set (1.34 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
datanode_id	node id	INTEGER
datanode_name	node name	STRING
datanode_type	0: master/slave; 1: MGR	INTEGER

2.1.33. show hotdb functions – show current available sharding function

This command is used to view the hotdb_function table in configdb. The statement is:

```
mysql> show hotdb functions;
```

Parameter description:

Parameter	Description	Type
pattern	optional, fuzzy query expression, match the key function_name	STRING
expr	optional, fuzzy query expression, match the key function_name	STRING

For example:

```
zy@192.168.200.2:(none) 5.7.19-HotDB-2.5.1 02:47:07> show hotdb functions;
+-------------+---------------------+---------------+----------------+
| function_id | function_name       | function_type | auto_generated |
+-------------+---------------------+---------------+----------------+
| 1           | test_route1         | ROUTE         | 0              |
| 3           | sp_mode1            | SIMPLE_MOD    | 0              |
| 5           | test_match1         | MATCH         | 0              |
| 7           | test_range1         | RANGE         | 0              |
| 9           | test_range2         | RANGE         | 0              |
| 15          | AUTO_GENERATE_CRC32 | AUTO_CRC32    | 1              |
| 19          | AUTO_CRC32_4        | AUTO_CRC32    | 0              |
| 21          | re                  | RANGE         | 0              |
| 25          | crc_mod1            | CRC32_MOD     | 0              |
| 27          | AUTO_GENERATE_MOD   | AUTO_MOD      | 1              |
| 31          | AUTO_CRC32_2        | AUTO_CRC32    | 0              |
| 33          | AUTO_CRC32_3        | AUTO_CRC32    | 0              |
| 36          | AUTO_CRC32_1        | AUTO_CRC32    | 0              |
| 37          | match1              | MATCH         | 0              |
+-------------+---------------------+---------------+----------------+
14 rows in set (0.01 sec)
```

For another example:

```
zy@192.168.200.2:(none) 5.7.19-HotDB-2.5.1 02:54:22> show hotdb functions like '%range%';
+-------------+---------------+---------------+----------------+
| function_id | function_name | function_type | auto_generated |
+-------------+---------------+---------------+----------------+
| 7           | test_range1   | RANGE         | 0              |
| 9           | test_range2   | RANGE         | 0              |
+-------------+---------------+---------------+----------------+
2 rows in set (0.01 sec)
```

```
zy@192.168.200.2:(none) 5.7.19-HotDB-2.5.1 02:54:49> show hotdb functions where function_name like '%range%';
+-------------+---------------+---------------+----------------+
| function_id | function_name | function_type | auto_generated |
+-------------+---------------+---------------+----------------+
| 7           | test_range1   | RANGE         | 0              |
| 9           | test_range2   | RANGE         | 0              |
+-------------+---------------+---------------+----------------+
2 rows in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
function_id	sharding function id	INTEGER
function_name	sharding function name	STRING
function_type	sharding type	STRING
auto_generated	whether it is the configuration auto-generated in HotDB or not (1: auto-generated, other: non-auto-generated)	INTEGER

2.1.34. show hotdb function infos – show current available sharding function information

This command is used to view the hotdb_function_info table in cofigdb. The statement is:

```
mysql> show hotdb function infos [WHERE expr];
```

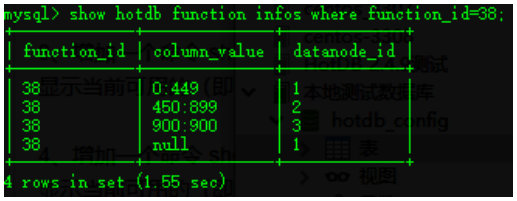
Parameter description:

Parameter	Description	Type
expr	optional: fuzzy query expression, match the specified key	STRING

For example:

```
zy@192.168.200.2:(none) 5.7.19-HotDB-2.5.1 02:48:52> show hotdb function infos;
+-------------+--------------+-------------+
| function_id | column_value | datanode_id |
+-------------+--------------+-------------+
| 1           | 0            | 9           |
| 1           | 1            | 11          |
| 1           | 2            | 13          |
| 1           | 3            | 15          |
| 1           | 4            | 9           |
| 1           | 5            | 11          |
| 1           | 6            | 13          |
| 1           | 7            | 15          |
| 1           | 8            | 9           |
| 1           | 9            | 11          |
| 3           | 0:6          | 9           |
| 3           | 14:20        | 13          |
| 3           | 21:27        | 15          |
| 3           | 28           | 0           |
| 3           | 7:13         | 11          |
| 5           | a            | 9           |
| 5           | B            | 11          |
| 5           | c            | 13          |
| 5           | d            | 15          |
| 5           | E            | 9           |
| 5           | f            | 11          |
| 5           | G            | 13          |
| 5           | H            | 15          |
```

For another example:



Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
function_id	sharding function id	INTEGER
column_value	sharding key value	STRING
datanode_id	data node id	INTEGER

2.1.35. show hotdb rules – show current available sharding function

This command is used to view the hotdb_rule table in configDB. The statement is:

```
mysql> show hotdb rules [LIKE 'pattern' | WHERE expr];
```

Parameter description:

Parameter	Description	Type
pattern	optional: fuzzy query expression, match the key rule_name	STRING
expr	optional: fuzzy query expression, match the key rule_name	STRING

For example:

```
zy@192.168.200.2:(none) 5.7.19-HotDB-2.5.1 02:48:58> show hotdb rules;
+---------+-----------------------------------------------+-------------+-------------+----------------+
| rule_id | rule_name                                     | rule_column | function_id | auto_generated |
+---------+-----------------------------------------------+-------------+-------------+----------------+
| 1       | AUTO_GENERATE_3_MATCH1_TB                     | ANAME       | 5           | 1              |
| 3       | AUTO_GENERATE_3_AUTO_TB                       | ID          | 15          | 1              |
| 7       | hotdb-cloud_fe269214-616a-4d7f-bbd8-620ae76a0bfc | a           | 19          | 0              |
| 9       | hotdb-cloud_241fe63c-e601-47e8-ab05-1634886855be | adnid       | 21          | 0              |
| 11      | AUTO_GENERATE_3_BULA2                         | BU          | 15          | 1              |
| 13      | AUTO_GENERATE_3_RANGE1_TB                     | ADNID       | 7           | 1              |
| 15      | AUTO_GENERATE_3_RANGE2_TB                     | ADNID       | 9           | 1              |
| 17      | AUTO_GENERATE_3_ROUTE1_TB                     | A           | 1           | 1              |
| 19      | AUTO_GENERATE_3_SP_MOD1_TB                    | ID          | 3           | 1              |
| 21      | AUTO_GENERATE_3_CRC_MOD2_TB                   | ID          | 25          | 1              |
| 23      | AUTO_GENERATE_3_FT_MOD1                       | ID          | 27          | 1              |
| 27      | hotdb-cloud_290f4c25-6f5e-4b93-86b9-2e41e59d0744 | name        | 19          | 0              |
| 29      | hotdb-cloud_720b4bff-6c89-4572-b1cc-b2ce5eb2dabd | name        | 19          | 0              |
| 31      | hotdb-cloud_46bafb4c-e613-43e5-bb0d-2ed49941639b | id          | 31          | 0              |
| 44      | hotdb-cloud_42a8e8b5-1922-480a-88fa-e56388746db2 | xuhao       | 36          | 0              |
| 46      | hotdb-cloud_7d291ff3-58ec-4794-8847-6a980208386e | id          | 19          | 0              |
+---------+-----------------------------------------------+-------------+-------------+----------------+
16 rows in set (0.01 sec)
```

For another example:

```
zy@192.168.200.2:(none) 5.7.19-HotDB-2.5.1 02:59:39> show hotdb rules like '%range%';
+---------+--------------------------+-------------+-------------+----------------+
| rule_id | rule_name                | rule_column | function_id | auto_generated |
+---------+--------------------------+-------------+-------------+----------------+
| 13      | AUTO_GENERATE_3_RANGE1_TB | ADNID       | 7           | 1              |
| 15      | AUTO_GENERATE_3_RANGE2_TB | ADNID       | 9           | 1              |
+---------+--------------------------+-------------+-------------+----------------+
2 rows in set (0.00 sec)
```

```
zy@192.168.200.2:(none) 5.7.19-HotDB-2.5.1 02:59:18> show hotdb rules where rule_name like '%range%';
+---------+--------------------------+-------------+-------------+----------------+
| rule_id | rule_name                | rule_column | function_id | auto_generated |
+---------+--------------------------+-------------+-------------+----------------+
| 13      | AUTO_GENERATE_3_RANGE1_TB | ADNID       | 7           | 1              |
| 15      | AUTO_GENERATE_3_RANGE2_TB | ADNID       | 9           | 1              |
+---------+--------------------------+-------------+-------------+----------------+
2 rows in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
rule_id	sharding function id	INTEGER
rule_name	sharding function name	STRING
rule_column	sharding key name	STRING
function_id	sharding type ID	INTEGER

auto_generated	whether it is the configuration auto-generated in HotDB or not (1: auto-generated, other: non-auto-generated)	INTEGER

2.1.36. show backupmasterdelay [DNID]– show master/slave replication delay of specified data node

The command is used to view the master/slave replication delay of specified data node[DNID], the statement is:

```
mysql> show backupmasterdelay [DNID];
```

Parameter description:

Parameter	Description	Type
DNID	Data node id	INTEGER

For example:

```
root@192.168.210.151:(none) 5.7.25 05:49:33> show backupmasterdelay 1;
+---------------+-----------+-----------------+-------------------+
| datasource_id | sql_delay | slave_io_running | slave_sql_running |
+---------------+-----------+-----------------+-------------------+
|             2 |         0 | Yes              | Yes               |
+---------------+-----------+-----------------+-------------------+
1 row in set (0.04 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
datasource_id	Data source id	INTEGER
sql_delay	Replication delay (s)	LONG
slave_io_running	Slave io_thread status (Yes/No)	STRING
slave_sql_running	Slave sql_thread status	STRING

	（Yes/No）	

2.2. HotDB services

2.2.1. show @@config_master_status - return to show master status of ConfigDB

This command is used to show the show master status of the current ConfigDB.

For example:

```
mysql> show @@config_master_status
```

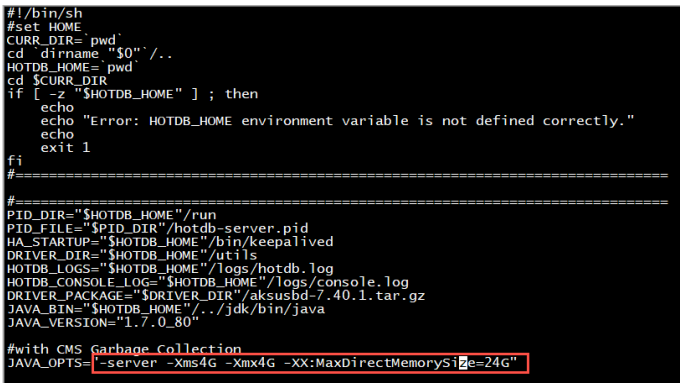
```
root@192.168.210.47:(none) 5.6.29-HotDB-2.5.5 10:12:25> show @@config_master_status;
+------------------+-----------+--------------+------------------+-------------------+
| file             | position  | binlog_do_db | binlog_ignore_db | executed_gtid_set |
+------------------+-----------+--------------+------------------+-------------------+
| mysql-bin.000344 | 311138178 |              |                  |                   |
+------------------+-----------+--------------+------------------+-------------------+
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
file	Binlog file	STRING
position	Binlog position	INTEGER
binlog_do_db	Database to be recorded by Binlog	STRING
binlog_ignore_db	Database to be ignorded by Binlog	STRING
executed_gtid_set	Executed GTID	STRING

2.2.2. show @@server – show the status of HotDB server

This command is used to show the running status of current HotDB Server. the memory conforms to the value in the configuration./bin/hotdb-server.



For example:

```
mysql> show @@server;
```

```
ct@127.0.0.1 : (none) 08:26:27> show @@server\G;
*************************** 1. row ***************************
           uptime: 7h 31m 51s
      online_time: 7h 31m 40s
      used_memory: 651M
     total_memory: 3584M
       max_memory: 3584M
 max_direct_memory: 24576M
used_direct_memory: 156M
      reload_time: 2018-05-30 12:54:38
          charset: utf8
             role: MASTER
           status: ON
             mode: READ-WRITE
          version: 2.4.9
1 row in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
uptime	the time of creating HotDB instance	STRING/[number“h” number“m” number”s”]
online_time	HotDB online time	STRING/[number“h” number“m” number”s”]

used_memory	the used memory	STRING/[number + “M”]
total_memory	the total memory	STRING/[number + “M”]
max_memory	the max memory	STRING/[number + “M”]
max_direct_memory	the max direct memory	STRING/[number + “M”]
used_direct_memory	the used direct memory	STRING/[number + “M”]
reload_time	Last re-load configuration time	STRING/[yyyy-MM-dd hh:mm:ss]
charset	charset	STRING/[charset]
role	the master/slave role	MASTER: master
		BACKUP: salve
status	the status of HotDB	ON: on
		OFF: off
mode	the read/write mode of HotDB	STRING/[“READ-ONLY”, “READ-WRITE”]
version	the version of HotDB	STRING/[number.number.number.number]

2.2.3. show @@serversourceusage – the usage of resources of current server

This command is used to view the usage of resources of current HotDB Server. For

example:

```
mysql> show @@serversourceusage;
```

```
root@127.0.0.1:(none) 5.1.27-HotDB-2.4.4 01:25:21> show @@serversourceusage \G
*************************** 1. row ***************************
 used_memory: 6855
total_memory: 128947
        disk: /dev/mapper/VolGroup-lv_root 53G 7.2G,/dev/sda1 500M 86M,/dev/mapper/VolGroup-lv_home 533G
    cpu_load: 0.56
   cpu_usage: 6.00,1.60,0.00,91.30,0.00,0.00,1.00,0.00
      net_in: 237679630
     net_out: 241084821
          io: sdb 0.76 0.30,sda 31.56 249.52
1 row in set (5.20 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
used_memory	used memory (MB)	STRING/[number]
total_memory	total memory (MB)	STRING /[number]
disk	usage of disk	STRING/[path number,…]
cpu_load	CPU load	FLOAT/[float]
cpu_usage	CPU usage rate	STRING/[number,number,…]
net_in	network flow rate (bytes/s)	LONG/[number]
net_out	network flow rate (bytes/s)	LONG/[number]
cores	total cores of CPU	INT[number]
io	disk read-write speed (kB/s)	STRING/[“sda” number number]

2.2.4. show @@systemconfig_memory - memory parameters of current compute node

This command is used to view the memory parameters usage of the current compute

node.

For example:

```
mysql> show @@systemconfig_memory;
```

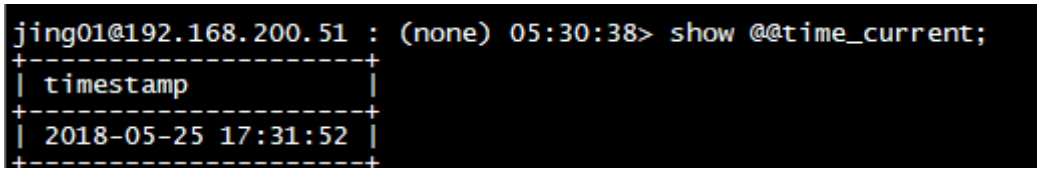
Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
config	configuration	STRING/[number]

2.2.5. show @@time_current – show the current time

This command is used to view the current time, for example:

```
mysql> show @@time_current;
```



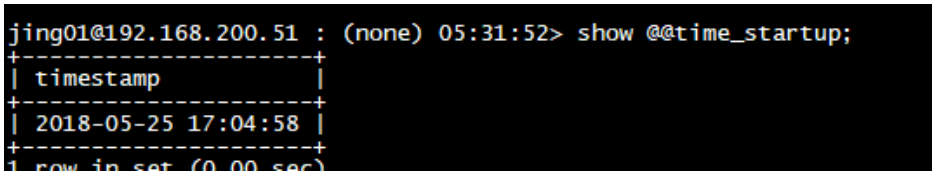
Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
timestamp	current time of HotDB Server	STRING/[yyyy-MM-dd HH:mm:ss]

2.2.6. show @@time_startup – show the startup time of HotDB

This command is used to view the startup time of HotDB Server. For example:

```
mysql> show @@time_startup;
```



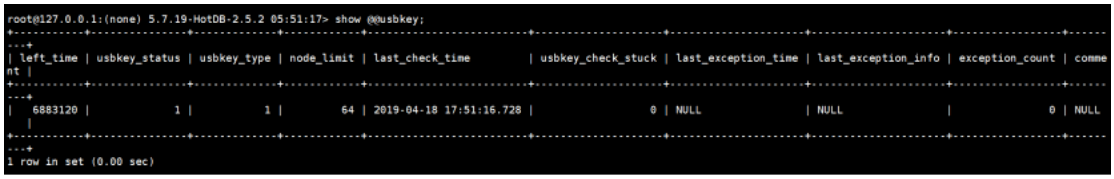
Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
timestamp	current time of HotDB Server	STRING/[yyyy-MM-dd HH:mm:ss]

2.2.7. show @@usbkey – show USB-KEY status

This command is used to show USB-KEY status (authorization) and detect the exception info of the authorization. For example:

```
mysql> show @@usbkey;
```



Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
left_time	left time (s)	LONG/[number]
usbkey_status	USB_KEY status	0: abnormal
		1: normal
usbkey_type	USB_KEY type	1: tentative
		2: terminable
		3: permanent
node_limit	limit of node number	INT/[number]
last_check_time	ending time of last detection	STRING/[yyyy-MM-dd HH:mm:sss]
usbkey_check_ stuck	whether detection of USB_KEY is stuck	0: not stuck
		1: stuck
last_exception _time	the time of last throwed exception in detection	STRING/[yyyy-MM-dd HH:mm:sss]
last_exception _info	the information of last throwed exception in detection	STRING
exception_count	the total times of last throwed exception in detection	INT/[number]

comment	comment	STRING

Note: left_time=0 means permanent or cancellation;

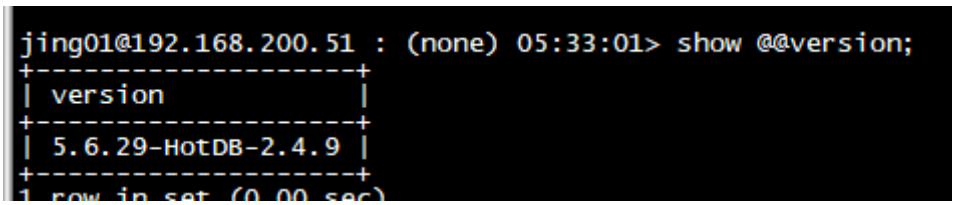
usbkey_check_stuck=1 means that thread is checked to be stuck. When the thread is checked to be stuck or the total times of last throwed exception in detection exceeds 1000, it prompts:

```
It is recommended to restart the HotDB server during the low peak period of business
```

2.2.8. show @@version – show USB-KEY status

This command is used to view the description of versions of HotDB Server. For example:

```
mysql> show @@version;
```



Fields and their description are contained in the result:

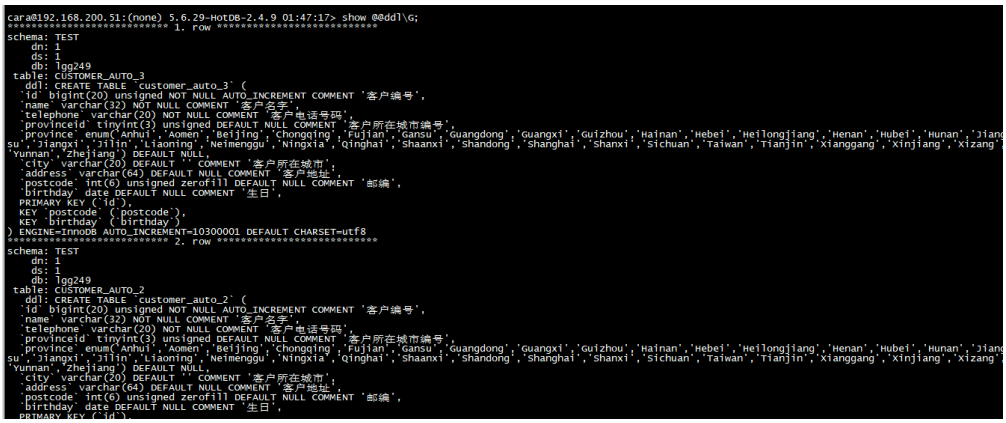
Column Name	Description	Value Type/Range
version	HotDB version	STRING

2.3. MySQL Services

2.3.1. show @@ddl – show DDL statements of tables

This command is used to show DDL statements of tables. For example:

```
mysql> show @@ddl;
```



Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
schema	LogicDB	STRING/[database]
dn	data node id	INT/[number]
ds	data source id	INT/[number]
db	database	STRING/[database]
table	table name	STRING/[table]
ddl	DDL statement of table	STRING/[sql]

2.3.2. show @@lastsql – the last executed sql of connection in borrowed status

This command is used to view the last executed SQL information of connection in borrowed status. For example:

```
mysql> show @@lastsql;
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
id	backend id	LONG/[number]
mysqlid	data node id	LONG/[number]
dn_ds	data node id – data source id	STRING/[number_number]
host	data source	STRING/[ip:port/database]
last_executed_sql	the last MySQL statement executed on the data source [host]	STRING/[sql]

2.3.3. show @@onlineddl – show the active onlineddl statement

This command shows the active OnlineDDL statement and its execution speed. The progress shows the execution progress of the statement by percentage. The speed shows the execution speed of current OnlineDDL statement (unit: row/ms). For example:

```
mysql> show @@onlineddl;
```

```
| schema    | onlineddl                                  | progress | speed   | table        | type |
+-----------+--------------------------------------------+----------+---------+--------------+------+
| TEST_PERF | ALTER TABLE CUSTOMER_Q_1 ADD COLUMN A INT |   0.2275 | 21.3297 | CUSTOMER_Q_1 |    1 |
+-----------+--------------------------------------------+----------+---------+--------------+------+
1 row in set (0.01 sec)

ct@127.0.0.1 : (none) 08:26:00> show @@onlineddl;
+-----------+--------------------------------------------+----------+---------+--------------+------+
| schema    | onlineddl                                  | progress | speed   | table        | type |
+-----------+--------------------------------------------+----------+---------+--------------+------+
| TEST_PERF | ALTER TABLE CUSTOMER_Q_1 ADD COLUMN A INT |   0.2625 | 32.1921 | CUSTOMER_Q_1 |    1 |
+-----------+--------------------------------------------+----------+---------+--------------+------+
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
schema	LogicDB	STRING/[database]
onlineddl	statement	STRING/[SQL]
progress	progress	LONG/[number]
speed	speed (row/ms)	LONG/[number]
table	table mane	STRING/[table]
type	change type	LONG/[number]

2.3.4. show @@tableindex – show index structure of tables

This command is used to show the index structure of each data table. For example:

```
mysql> show @@tableindex;
```

```
root@127.0.0.1 : mydb 03:43:25> show @@tableindex ;
+--------+------+------+------+-------+-----------------+
| schema | dn   | ds   | db   | table | index           |
+--------+------+------+------+-------+-----------------+
| MYDB   | 3    | 3    | db03 | mytb  | 1 id_index 1 id |
| MYDB   | 2    | 2    | db02 | mytb  | 1 id_index 1 id |
| MYDB   | 1    | 1    | db01 | mytb  | 1 id_index 1 id |
+--------+------+------+------+-------+-----------------+
3 rows in set (0.51 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
schema	LogicDB	STRING/[database]
dn	data node id	INT/[number]
ds	data source id	INT/[number]
db	database	STRING/[database]
table	database name	STRING/[number]
index	table index structure	STRING

2.4. Sharding plan online modification

This section introduces the command of online modifying sharding plan in Management Port, with the following steps:

- Step one, check the modification plan (onlinemodificationrulecheck)

- Step two, modify the sharding plan (onlinemodificationrule)
- Step three, view the modification progress (onlinemodificationruleprogress)
- Whether to continue the modification (onlinemodificationrulecontinue). Cancelling the current ongoing task (onlinemodificationrulecancel) is an optional operation item and can be executed according to actual situation

The above steps shall be carried out in sequence, or the result may not be successful.

2.4.1. onlinemodificationrulecheck

This command is used to check the sharding plan online modification. For example:

```
onlinemodificationrulecheck db.tablename[=functionid,rulecol:datanodes:checkconsistency (whether to check the master/slave consistency 1|0)]
```

The command contains the following fields and their description:

Parameter	Description
db	LogicDB
tablename	table name
functionid	refer to the table hotdb_function in the configdb hotdb_config for the sharding rule id
rulecol	sharding key
datanodes	refer to the table hotdb_datanode in the configdb hotdb_config for the data node
checkconsistency	whether to check the master/slave consistency 1|0

It is used in two ways:

1. It can be used to check whether sharding rule modification related items pass. The

check item id and corresponding check items are as follows:

Check Item ID	Corresponding Key	Description of Check Item
1	tbNameLess45	The source table name is not longer than 45 characters
2	running	No sharding plan modification task is being executed in source table
3	validCol	Sharding key is the key contained in table structure
4	diffrule	The sharding function and sharding key in modification plan are inconsistent with those in source table
5	existUniqueKey	The source table has the master key or unique key
6	recommendColType	The sharding key is a type of key recommended by the current sharding function
7	lostData	The new sharding plan will not result in data loss
8	trigger	The source table has no trigger
9	refByTrigger	The source table is not associated with other triggers
10	foreignConstraint	The source table has no foreign key constraint

11	consistency	The master/slave data consistency check result of the source table is consistent

If the check result (result value) is 1, it means that the check of this item fails, and the modification result may be incorrect.

As shown below: cpd_test is LogicDB, zx_cvset_signin_result is table name, 4 is functionid, id is sharding key, [1,2] is data node, 1 is master/slave consistency check.

```
mysql> onlinemodificationrulecheck cpd_test. zx_cvset_signin_result=4,id:1,2:1;
```

```
zy@192.168.200.51 : (none) 02:34:12> onlinemodifyrulecheck cpd_test.zx_cvset_signin_result=4,id:1,2:1;
+----------+------------------------+------+--------+---------+
| db       | tablename              | id   | result | warning |
+----------+------------------------+------+--------+---------+
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    1 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    2 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    3 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    4 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    5 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    6 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    7 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    8 |     -1 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    9 |     -1 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |   10 |     -1 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |   11 |     -1 |    NULL |
+----------+------------------------+------+--------+---------+
11 rows in set (0.02 sec)
```

When multi tables are checked at the same time, information between tables shall be separated by spaces. For example:

```
onlinemodificationrulecheck db.tablename=functionid,rulecol:datanodes:checkconsistency [db.tablename=functionid,rulecol:datanodes:checkconsistency..]
```

```
zy@192.168.200.51 : (none) 10:55:17> onlinemodifyrulecheck  cpd_test.cv_live_courseware=4,id:1,2,3,4:1 cpd_test.cv_live_study=3,id:1,2,3,4,5,6,7,8:1;
+----------+--------------------+------+--------+---------+
| db       | tablename          | id   | result | warning |
+----------+--------------------+------+--------+---------+
| CPD_TEST | CV_LIVE_COURSEWARE |    1 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    2 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    3 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    4 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    5 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    6 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    7 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    8 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    9 |     -1 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |   10 |     -1 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |   11 |     -1 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    1 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    2 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    3 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    4 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    5 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    6 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    7 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    8 |     -1 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    9 |     -1 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |   10 |     -1 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |   11 |     -1 |    NULL |
+----------+--------------------+------+--------+---------+
22 rows in set (0.05 sec)
```

2. It can be used to view the check result after the sharding plan online modification is

checked. For example:

```
onlinemodificationrulecheck db.tablename [db.tablename…]
```

Fields and their description are contained in the result:

Column Name	Description
db	LogicDB
tablename	table name
id	check item id
result	result (0 pass, 1 fail, -1 checking)
warning	error warning message

View whether the check is finished (if the result value is -1, it means that the check is not finished), or whether there is any item that fails to pass the check (if the result value is 1, it means that items fail to pass the check), if all the result values are 0, it means that the sharding plan can be modified.

As shown below: cpd_test is LogicDB, zx_cvset_signin_result is table name.

```
zy@192.168.200.51 : (none) 10:29:24> onlinemodifyrulecheck cpd_test.zx_cvset_signin_result;
+----------+-----------------------+------+--------+---------+
| db       | tablename             | id   | result | warning |
+----------+-----------------------+------+--------+---------+
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    1 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    2 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    3 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    4 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    5 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    6 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    7 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    8 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |    9 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |   10 |      0 |    NULL |
| CPD_TEST | ZX_CVSET_SIGNIN_RESULT |   11 |      0 |    NULL |
+----------+-----------------------+------+--------+---------+
11 rows in set (0.00 sec)
```

When check results of multi tables are viewed at the same time, tables shall be separated by spaces. For example:

```
zy@192.168.200.51 : (none) 10:55:53>  onlinemodifyrulecheck  cpd_test.cv_live_courseware cpd_test.cv_live_study;
+----------+--------------------+------+--------+----------+
| db       | tablename          | id   | result | warning |
+----------+--------------------+------+--------+----------+
| CPD_TEST | CV_LIVE_STUDY      |    1 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    2 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    3 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    4 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    5 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    6 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    7 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    8 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |    9 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |   10 |      0 |    NULL |
| CPD_TEST | CV_LIVE_STUDY      |   11 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    1 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    2 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    3 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    4 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    5 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    6 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    7 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    8 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |    9 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |   10 |      0 |    NULL |
| CPD_TEST | CV_LIVE_COURSEWARE |   11 |      0 |    NULL |
+----------+--------------------+------+--------+----------+
22 rows in set (0.00 sec)
```

2.4.2. onlinemodificationrule

This command is used to modify the sharding plan. Execute the command to return to the result: OK or MySQLException.

onlinemodificationrule db.tablename=functionid,rulecol:datanodes: source table handling (hour:0 means perserved): batch row (1000): replication interval (T3/I0.3): waiting timeout (days): pause data replication period;

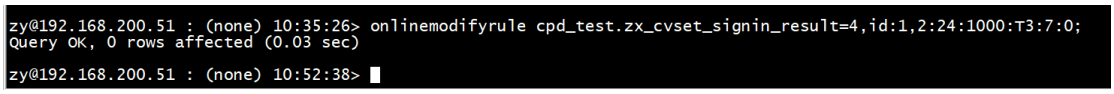
Fields and their description are contained in the command:

Parameter	Description
db	LogicDB
tablename	table name
functionid	refer to the table hotdb_function in the configdb hotdb_config for the sharding rule id
rulecol	sharding key
datanodes	refer to the table hotdb_datanode in the configdb hotdb_config for the data node
source table	source table handling mode after successful

handling	sharding plan (preserved for n hours, o means not preserved)
batch row	limit the read/write row size in data replication phase each time
replication interval	The interval time of read/write row each time (T3: 3 times of SQL execution time, I0.3: fixed time 0.3s)
waiting timeout	The time of waiting for user to handle the data inconsistency caused by the modification. If the user does not confirm it over the set time, the modification task will automatically fail, set range [1,30]
pause data replication period	The data replication of the modificaiton task is automatically paused in the set time range, and the time interval is separated by commas. For example: 0700-2210,0300-0559

As shown below: cpd_test is LogicDB, zx_cvset_signin_result is table name, 4 is functionid, id is sharding key, [1,2] is data node, 24 means source table will be deleted after 24 hours, 1000 is batch row, T3 is 3 times of SQL execution time, 7 means waiting timeout of 7 days, 0 means not set period of pause data replication

```
onlinemodificationrule cpd_test. zx_cvset_signin_result=4,id:1,2:24:1000:T3:7:0;
```



When multi tables are modified at the same time, the tables shall be separated by spaces.

```
zy@192.168.200.51 : (none) 10:59:21> onlinemodifyrule cpd_test.cv_live_courseware=4,id:1,2,3,4,:0:1000:T3:7: cpd_test.cv_live_study=3,id:1,2,3
,4,5,6,7,8:0:1000:T3:7:;
Query OK, 0 rows affected (0.04 sec)

zy@192.168.200.51 : (none) 10:59:46> onlinemodifyruleprogress cpd_test.cv_live_courseware,cpd_test.cv_live_study;
+---------+--------------------+----------+------+--------+--------+------+------+------------+------------+
| db      | tablename          | progress | cost | state  | detail | lost | over | inconsitent | autorepair |
+---------+--------------------+----------+------+--------+--------+------+------+------------+------------+
| CPD_TEST | CV_LIVE_COURSEWARE |      100 |  737 | FINISH | NULL   | null | null | null       |       null |
| CPD_TEST | CV_LIVE_STUDY      |      100 |  737 | FINISH | NULL   | null | null | null       |       null |
+---------+--------------------+----------+------+--------+--------+------+------+------------+------------+
2 rows in set (0.00 sec)
```

The functionid used for sharding plan modification can be viewed in the table hotdb_function in configdb.

```
root@localhost : hotdb_config_cp_249wjw 11:37:47> select * from hotdb_function;
+-------------+---------------+---------------+----------------+
| function_id | function_name | function_type | auto_generated |
+-------------+---------------+---------------+----------------+
|           1 | AUTO_4        | AUTO          |              0 |
|           2 | AUTO_3        | AUTO          |              0 |
|           3 | AUTO_MOD_8    | AUTO_MOD      |              0 |
|           4 | AUTO_CRC32_4  | AUTO_CRC32    |              0 |
|           5 | AUTO_CRC32_2  | AUTO_CRC32    |              0 |
|           6 | AUTO_2        | AUTO          |              0 |
+-------------+---------------+---------------+----------------+
6 rows in set (0.00 sec)
```

Special instructions:

- The functionid required for sharding rule modifiction has already existed in the table hotdb_function in configdb;
- When using the sharding rule to be modified, we must ensure that the number of data nodes specified is consistent with that of data nodes in function_id;
- The source table must have master key or unique key, no trigger and no foreign key constraint, or the modification result may be incorrect;
- If the parameter of source table handling is 0 when in modification, then the historical table will be preserved in the table information, and the naming format is: “source table name + roYYMMDDHHMMSS”;
- If the modification of sharding rule of a table in the same batch fails, then the modification of all tables in this batch will fail;
- The sever cannot be restarted when executing this command, or the modification may fail, but the original table may be preserved.

2.4.3. onlinemodificationruleprogress

This command is used to view the sharding plan modification progress, a table will have a row of data, as shown below:

```
onlinemodificationruleprogress db.tablename[,db1.tablename1,..]
```

Fields and their description are contained in the command:

Parameter	Description
db	LogicDB
tablename	table name

As shown below: cpd_test is LogicDB, cv_live_courseware and cv_live_study are table names.

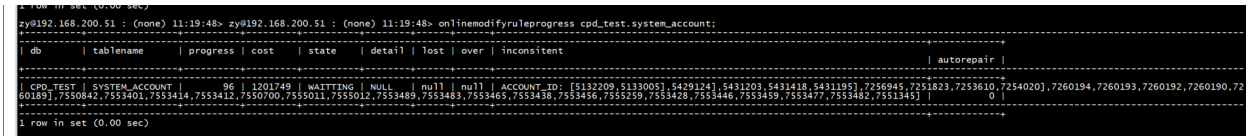
```
zy@192.168.200.51 : (none) 10:59:46> onlinemodifyruleprogress cpd_test.cv_live_courseware,cpd_test.cv_live_study;
+----------+--------------------+----------+------+--------+--------+------+------+-------------+-------------+
| db       | tablename          | progress | cost | state  | detail | lost | over | inconsitent | autorepair |
+----------+--------------------+----------+------+--------+--------+------+------+-------------+-------------+
| CPD_TEST | CV_LIVE_COURSEWARE |      100 |  737 | FINISH | NULL   | null | null | null        |       null |
| CPD_TEST | CV_LIVE_STUDY      |      100 |  737 | FINISH | NULL   | null | null | null        |       null |
+----------+--------------------+----------+------+--------+--------+------+------+-------------+-------------+
2 rows in set (0.00 sec)
```

Fields and their description are contained in the result:

Key Name	Description
db	LogicDB
tablename	table name
progress	0-100, integer
cost	execution duration (ms)
state	stopping (non-execution window), running (is being executing), waiting (inconsisitent, waiting

	for the user to confirm whether to continue), finish (completed), error (fail)
detail	other errors
lost	data lost
over	data over
inconsitent	data inconsistent
autorepair	autorepair (1/0): 1 means repaired, o means unrepaired

If the state returns to waitting, the user needs to confirm whether to continue it, ignore the inconsistent data or cancel the modification.



2.4.4. onlinemodificationrulecontinue

When the state of sharding plan modification progress is WAITTING, and the returned data is inconsistent, this command can be used to continue the modification, as shown below:

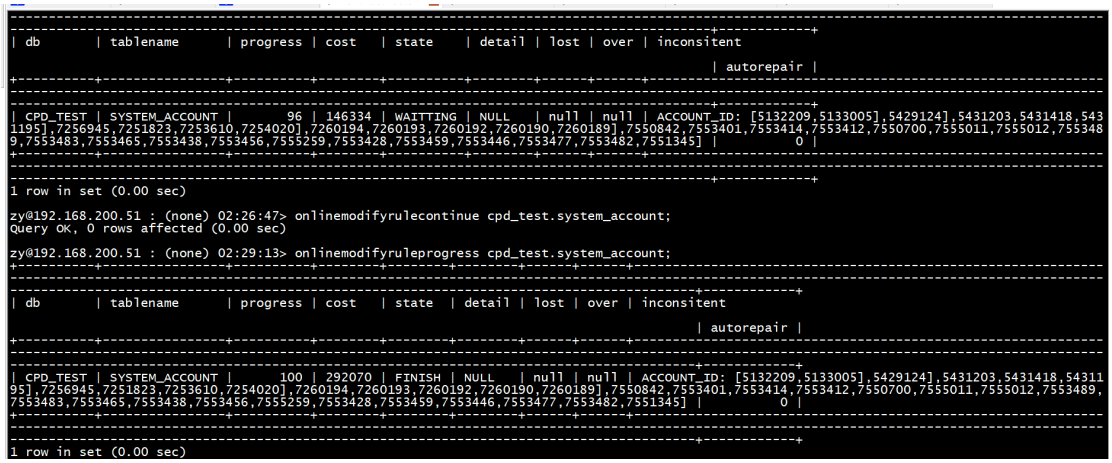
```
onlinemodificationrulecontinue db.tablename;
```

fields and their description are contained in the command:

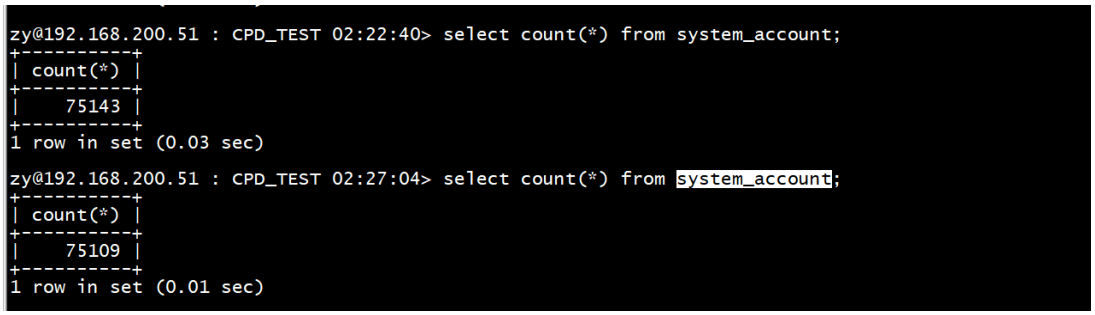
Parameter	Description
db	LogicDB

tablename	table name

As shown below: in the process of modifying sharding plan, when the state is waitting, and inconsistent data exits, this command is used to continue the modification, and then view the progress again, the progress is 100 and the state is finish.



Ignoring these inconsistent data may lead to data errors, as shown below: some data is lost after modification.



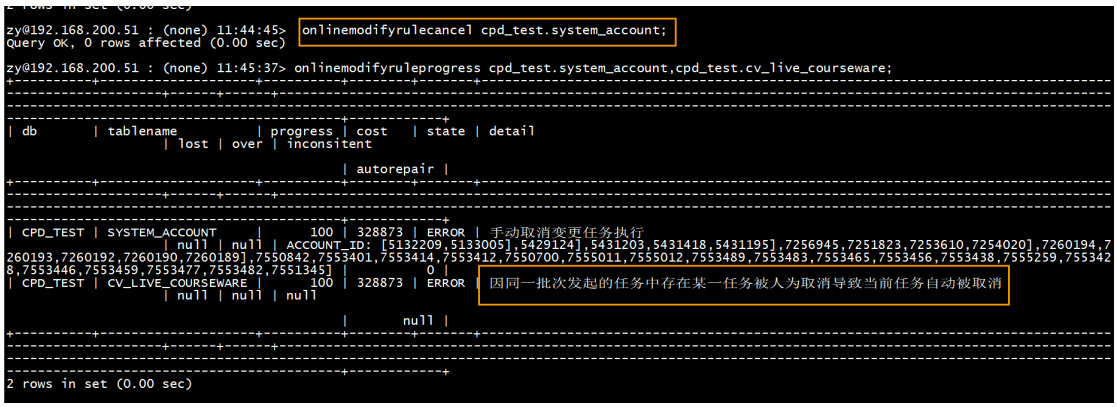
2.4.5. onlinemodificationrulecancel

This command is used to cancel the current ongoing task:

```
onlinemodificationrulecancle db.tablename;
```

If the modification of a table in the same batch is cancelled, then the modification of

sharding plan of all tables in this batch will be cancelled, as shown below:



3. Management Control Statement

3.1. check @@datasource_config – Check MySQL parameter configuration information

This command is used to check the consistency of particular parameter configuration information of MySQL data source. If any parameter is different from the requirements of HotDB Server, the system will output a prompt message after executing the command. For example:

```
mysql> check @@datasource_config;
```

```
ct@192.168.200.51 : (none) 01:50:07>  Check @@datasource_config;
+-------+-------+-----------------------------------------------------------------------------------------------------------------------------------+-----------------+
| Level | Code  | Messege                                                                                                                           | Value           |
+-------+-------+-----------------------------------------------------------------------------------------------------------------------------------+-----------------+
| Error | 10025 | DataNode=dn05,Datasource=192.168.200.52_3312_db249,ip=192.168.200.52,port=3312,Database=db249,type=主库    | autocommit=ON   |
| Error | 10025 | DataNode=dn03,Datasource=192.168.200.51_3310_db249,ip=192.168.200.51,port=3310,Database=db249,type=主库    | autocommit=OFF  |
| Error | 10025 | DataNode=dn04,Datasource=192.168.200.51_3311_db249,ip=192.168.200.51,port=3311,Database=db249,type=主库    | autocommit=ON   |
| Error | 10025 | DataNode=dn01,Datasource=192.168.200.51_3308_db249,ip=192.168.200.51,port=3308,Database=db249,type=主库    | autocommit=ON   |
| Error | 10025 | DataNode=dn06,Datasource=192.168.200.51_3313_db249,ip=192.168.200.51,port=3313,Database=db249,type=主库    | autocommit=ON   |
| Error | 10025 | DataNode=dn02,Datasource=192.168.200.51_3309_db249,ip=192.168.200.51,port=3309,Database=db249,type=主库    | autocommit=ON   |
+-------+-------+-----------------------------------------------------------------------------------------------------------------------------------+-----------------+
6 rows in set (0.02 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
Level	exception information level (Warning, Error)	STRING/[“Error”,“Warning”]
Code	exception code	INT/[number]
Message	error message	STRING
Value	error or warning related value	STRING

The parameters or configuration below require that all data sources shall be consistent and conform to the parameter configuration standard:

```
The time difference between data source and HOTDB is less than 3s

Read-only
```

Read-only

Completion_type must be NO_CHAN

Div_precision_increment

Div_precision_increment

Innodb_rollback_on_timeout must be ON

Autocommit

Tx_isolation

MAX_ALLOWED_PACKET

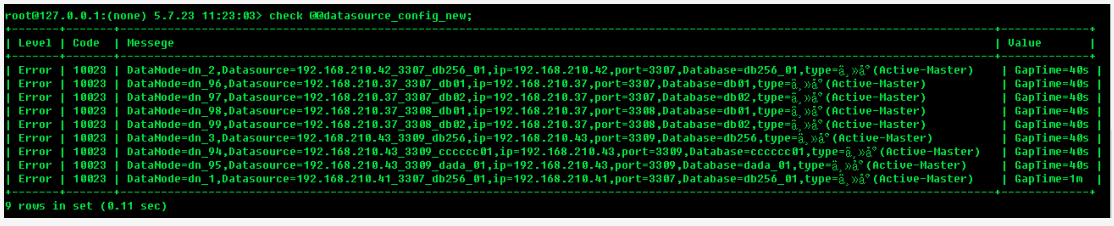
Please refer to the chapter MySQL Server Parameter Check in Distributed Transaction Database HotDB Server-V2.5.2 [Standard] Function Manual V1.0 for the detailed use methods and requirements.

3.2. check @@datasource_config_new – Check MySQL parameter configuration information

The function of this command is similar to check @@datasource_config. The difference is:

check @@datasource_config_new is to read and check data node information from the non-running table without recording the check status history.

mysql> check @@datasource_config_new;



Please refer to check @@datasource_config for usage and instructions.

3.3. check @@route – Route check

This command is used to check the rightness of sharding table data routing. The statement is:

```
mysql> check @@route [db_name.tb_name | tb_name];
```

Parameter description:

Parameter	Description	Type
db_name	database name	STRING
tb_name	table name	STRING

When the data routing is consistent, the result is:

```
jing01@192.168.200.51 : (none) 05:08:22> check @@route lgg.aa;
Empty set (0.04 sec)
```

When the data routing is inconsistent, the result is:

```
ct@127.0.0.1 : (none) 01:32:02> check @@route test_jzl.borrower;
+-----------------+----------+-----------+
| shard_key_value | route_dn | actual_dn |
+-----------------+----------+-----------+
| jones           | 1        | 3         |
+-----------------+----------+-----------+
1 row in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range

shard_key_value	the routing key value	STRING
route_dn	the routing node	INT/[number]
actual_dn	the actually stored node	INT/[number]

3.4. kill @@connection – Close a specified connection

This command is used to close the specified frontend connection, multi connections can be closed at the same time. The statement is:

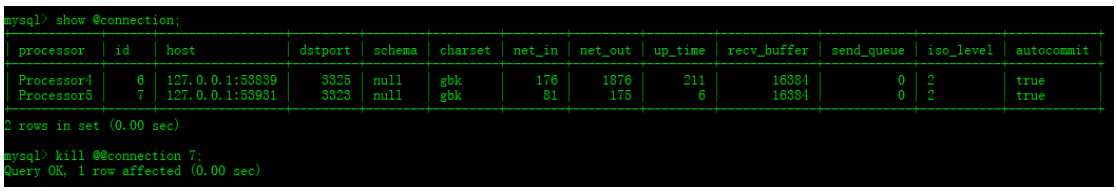
```
mysql> kill @@connection [id1,id2,id3…idn];
```

Parameter description:

Parameter	Description	Type
connection_id	the connected id	INTEGER/obtained through the command [show @connection]

For example:

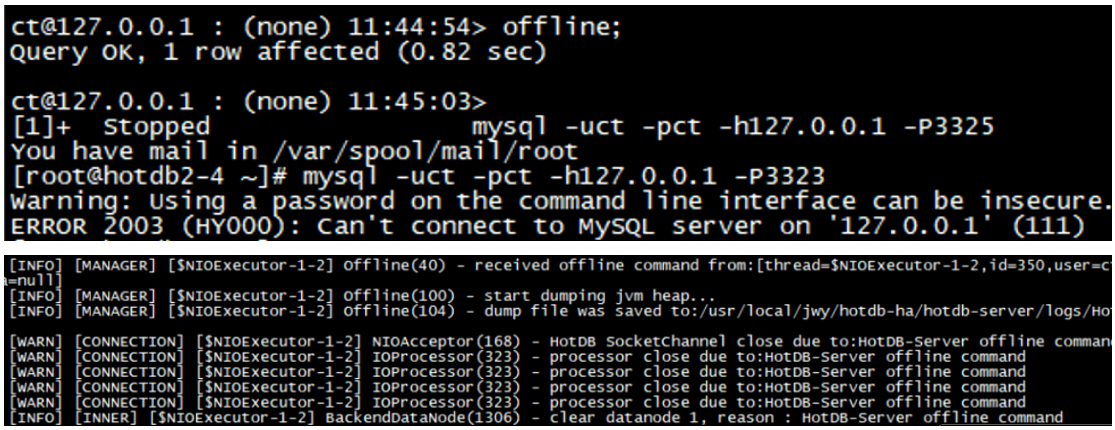
```
mysql> kill @@connection 7;

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



3.5. offline – HotDB offline

This command is used to close the HotDB Server port and disconnect the frontend connection of server port 3323. For example:

```
mysql> offline;
```



3.6. online – HotDB online

If we need to start up the HotDB Server port, we need to run the online in management end. This command is used to start up the HotDB Server port or in scenario where high availability switch occurs. The statement is:

```
mysql> online;
```

```
ct@127.0.0.1 : (none) 11:48:44> online;
Query OK, 1 row affected (5.51 sec)

ct@127.0.0.1 : (none) 11:48:52>
[2]+  Stopped                 mysql -uct -pct -h127.0.0.1 -P3325
You have mail in /var/spool/mail/root
[root@hotdb2-4 ~]# mysql -uct -pct -h127.0.0.1 -P3323
Warning: Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1606
Server version: 5.7.19-HotDB-2.4.9 HotDB Server by Hotpu Tech

Copyright (c) 2000, 2016, 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.

ct@127.0.0.1 : (none) 11:49:07>
```

```
[INFO] [MANAGER] [$NIOExecutor-2-0] Online(78) - received online command from:[thread=$NIOExecutor-2-0,id=1600,user=ct,host=127.0.0
null]
[INFO] [INIT] [$NIOExecutor-2-0] HotdbConfig(708) - using config datasource in start up:[id:-1,nodeId:-1 192.168.200.52:3306/hotdb_

[INFO] [INIT] [$NIOExecutor-2-0] HotdbConfig(340) - master config datasource [id:-1,nodeId:-1 192.168.200.52:3306/hotdb_config_ha_2
ccess.
[INFO] [INIT] [$NIOExecutor-2-0] BackendDatasource(600) - init datasource success,initSize= 8,success=8,ds=[id:-1,nodeId:-1 192.168
tus:1,charset:utf8]
[INFO] [INIT] [$NIOExecutor-2-0] HotdbServer(583) - init hotdb_config datanode success.
[INFO] [INIT] [$NIOExecutor-2-0] HotdbServer(587) - initialize data nodes ...
```

In a complete and normal HotDB Server high availability environment, if we manually send the command online to slave compute node, the slave compute node may start up 3323, and send the command offline to master compute node, and then the master compute node server port 3323 will be closed. But in the current state, the vip of keepalived will not drift (because master compute port 3325 is still available), then the compute node data service may become unavailable virtually. Therefore, if users manually operate the online of slave compute node without knowing clearly the operation mode of hign availability system or the existence of this defect, there may be a high risk of business failure!

3.7. 1.1. online_dr - switch the IDC

The master compute node of DR center is not involved in HA high availability switching of master center. In addition to some show commands, only this command is accepted: online_dr switch the IDC.

```
root@192.168.220.183:(none) 8.0.15-HotDB-2.5.3.1 08:12:31> online_dr;
```

```
Query OK, 1 row affected (5 min 4.35 sec)
```

When the IDC-level switching of the compute node occurs, that is, when the master compute node of the DR center provides services, if the master compute node of the DR center fails at this time, it can execute enable_online; command before executing online or online_dr command to enable the slave compute node of the DR center. At this time, the slave compute node in the DR center can automatically enable the service port (3323 by default) to continue the service.

```
root@192.168.220.184:(none) 8.0.15-HotDB-2.5.3.1 08:10:31> enable_online;

Query OK, 1 row affected (11 min 5.39 sec)

root@192.168.220.184:(none) 8.0.15-HotDB-2.5.3.1 08:22:27> online_dr;

Query OK, 1 row affected (0.01 sec)
```

3.8. rebuild @@pool – Rebuild current available data source of all nodes

This command is used to rebuid the backend connection of current HotDB Server and the connection information of data source. The statement is:

```
mysql> rebuild @@pool;

Query OK, 1 row affected (0.24 sec)
```

3.9. reload @@config – Reread configuration information

This command is used to upgrade configuration, such as upgrading the configuration of the file server.xml and memory. Enter this command in the command window, the configuration parameter can be upgraded without restarting HotDB Server. This command is the same as the dynamic loading of management platform. The result is as following: # this command is not suitable for all parameters

```
mysql> reload @@config;

Query OK, 1 row affected (2.31 sec)

Reload config success
```

3.10. reset @@reloading – Release the ongoing reload status by force

This command is used to release the ongoing reload status by force, that is, manually cancel the ongoing dynamic loading by force. Note: this command can be executed only when you confirm that there is no effect at all and this command can be used to reset dynamic loading when it is stuck. It is not recommended to use this command in any other circumstance.

For example, when dynamic loading is stuck:

```
root> reload @@config;

…stuck and no returned result…
```

Then if you confirm that the forced release of the ongoing reload status has no effect, you can execute this command to cancel the dynamic loading by force:

```
root> reset @@reloading;

Query OK, 1 row affected (0.00 sec)

Reset reloading success
```

Then the previous state that dynamic loading is stuck will be cancelled, and the following information will be shown:

```
root> reload @@config;

ERROR 1003 (HY000): Reload config failure, Reloading was set to false manually.
```

And the compute node log record is as follow:

```
2019-07-19 17:49:57.626 [WARN] [MANAGER] [$NIOExecutor-3-0] ResetHandler(27) - received reset @@reloading from [thread=$NIOExecutor-3-0,id=780,user=re,host=127.0.0.1,port=2475,localport=28613,schema=null], reloading will be set to false.
```

```
2019-07-19 17:50:04.336 [WARN] [MANAGER] [Labor-181] HotdbConfig(1331) - Reload config failure, Reloading was set to false manually.
```

3.11. restart @@heartbeat – Restart the heartbeat detection on the specified data node

This command is used to restart the heartbeat detection on the specified data node of the specified node. The statement is:

```
mysql> restart @@heartbeat [datanode_id];
```

Parameter description:

Parameter	Description	Type
datanode_id	data node id	INT

For example:

```
mysql> restart @@heartbeat 1;

# Restart the heartbeat detection function of node 1

Query OK, 2 rows affected (0.00 sec)
```

3.12. stop @@heartbeat – Stop the heartbeat on the specified data node for a period

This command is used to stop the heartbeat on the specified data node for a period. When the time is -1, system will cancel the stop state of the specified node. The statement is:

```
mysql> stop @@heartbeat [datanode_id:time(s)]
```

Parameter description:

Parameter	Description	Type
datanode_id	data node id	INT
time	stop time (s)	INT

For example:

```
mysql> stop @@heartbeat 1:60;

# Stop node 1 for 60s

Query OK, 1 row affected (0.01 sec)



mysql> stop @@heartbeat 1:-1;

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

3.13. switch @@datasource – Switch the specified data source to standby data source

This command is used to switch the data source of the specified data node to the next standby data source. The statement is:

```
mysql> switch @@datasource [datanode_id];
```

For example:

```
mysql> stop @@heartbeat 1:60;

# Stop node 1 for 60s

Query OK, 1 row affected (0.01 sec)
```

```
mysql> stop @@heartbeat 1:-1;

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

4. Control statements related to IDC switching in DR mode

The commands described in this chapter only need to be known by users. They are mainly used for interactive judgment between the management platform and the compute node service in the process of IDC switching. In daily use, manual call of these commands is forbidden.

4.1. disable_election – Disable election in cluster

```
mysql> disable_election;

Query OK, 1 row affected (0.01 sec)
```

Generally used when switching the IDC in the DR mode, this command is used to disable election in the internal cluster of the IDC that is providing services, so as to avoid the occurrence of IDC switching and possible affect on the final result of the IDC switching.

4.2. enable_election – Enable election in cluster

```
mysql> enable_election;

Query OK, 1 row affected (0.01 sec)
```

4.3. disable_non_query_command – Only allow query command

```
mysql> disable_non_query_command;
```

```
Query OK, 1 row affected (0.01 sec)
```

This command is the internal command when switching the IDC in the DR mode. Once called, the instance of the compute node will only allow query command, and non-query command will be released after the switching is successful.

4.4. enable_non_query_command – Allow non-query command

```
mysql> enable_non_query_command;

Query OK, 1 row affected (0.01 sec)
```

4.5. offline_to_dr – Execute offline and online is not allowed

```
mysql> offline_to_dr;

Query OK, 1 row affected (0.01 sec)
```

4.6. exchangeconfig – Exchange configuration of IDC

```
mysql> exchangeconfig;

Query OK, 1 row affected (0.01 sec)
```

4.7. exchangememoryconfig – Exchange configuration in memory

```
mysql> exchangememoryconfig;
```

```
Query OK, 1 row affected (0.01 sec)
```

4.8. online_dr_check – Check IDC switching

```
mysql> online_dr_check;

Query OK, 1 row affected (0.01 sec)
```

4.9. online_dr_process – Show IDC switching process

This command is used to view the progress of IDC switching in the DR mode, for example:

```
root@192.168.210.78:(none) 8.0.15 06:51:52> ONLINE_DR_PROCESS;
+---------+-------+------------+--------+
| process | error | error_code | status |
+---------+-------+------------+--------+
|       0 |  null |          0 |      1 |
|       1 |  null |          0 |      1 |
|       2 |  null |          0 |      1 |
|       3 |  null |          0 |      1 |
|       4 |  null |          0 |      1 |
|       5 |  null |          0 |      1 |
|       6 |  null |          0 |      1 |
|       7 |  null |          0 |      1 |
|       8 |  null |          0 |      1 |
+---------+-------+------------+--------+
9 rows in set (0.00 sec)
```

Fields and their description are contained in the result:

Column Name	Description	Value Type/Range
process	Handling process, 0-8	INTEGER
error	Error message (error format: srcDs1:dstDs1,srcDs2:dstDs2,...;errormsg or ds,ds:ds,...;errormsg, data source	STRING

	format (datanodeID_datasourceID_datasourceIP_port_dbname)，if included：then src is the original master center，dst is the original DR center)	
error_code	Error code status, 1 is finished, 0 is unfinished	INTEGER
status	Status, 1 is finished, 0 is unfinished	INTEGER

4.10. reset dberrorcount – Clear all the error messages of LogicDBs

```
mysql> reset dberrorcount;

Query OK, 1 row affected (0.01 sec)
```

5. Function Processing Statement

5.1. dbremapping @@add@ - Add database mapping relation

This command is used to add the database mapping relation. The statement is:

```
mysql> dbremapping @@add@[database_name]:[database_name],[database_name]:[database_name]…;
```

For example:

```
mysql> dbremapping @@add@db01:logic_db01,db02:logic_db02;

# Add multiple mapping relations

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

Add mapping relations from the database db01 to the LogicDB logic_db, so that executing the SQL statement USE db01 is equivalent to executing USE logic_db:

```
mysql> dbremapping @@add@db01:logic_db;
```

Note: if you add mapping relation from the same name database to different LogicDB, the previous mapping relations will be overlaid. Mapping relations are allowed to be added from different databases to the same LogicDB.

For example, first add the mapping relation from db01 to logic_db01:

```
dbremapping @@add@db01:logic_db01
```

And then add the mapping relation from db01 to logic_db02:

```
dbremapping @@add@db01:logic_db02
```

The first command will be overlaid by the second command, that is, only the mapping relation from db01 to logic_db02 exists finally. You can view the existing mapping

relations in the line dbremapping of the table hotdb_config_info in compute node ConfigDB:

```
hotdb_config@127.0.0.1:hotdb_config_test252 5.7.23-log 05:01:56> select * from hotdb_config_info\G
*************************** 1. row ***************************
          k: dbremapping
          v: DB254:MY_DB,DB253:YOU_DB,DB252:MY_DB
description: NULL
```

5.2. dbremapping @@remove@ - Remove database mapping relation

This command is used to remove the database mapping relations added in dbremapping @@add@. The statement is:

```
mysql> dbremapping @@remove@[database_name]:[database_name],[database_name]:[database_name]…;
```

For example:

```
mysql> dbremapping @@remove@db01:logic_db01,db02:logic_db02;

#remove multiple mapping relations

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

5.3. onlineddl – OnlineDDL operation

This command ensures that when modifying the data table structure, the read/write of online business will not be blocked, and database can still provide normal data access service. The statement is:

```
mysql> onlineddl "[DDLSTATEMENT]";
```

For example:

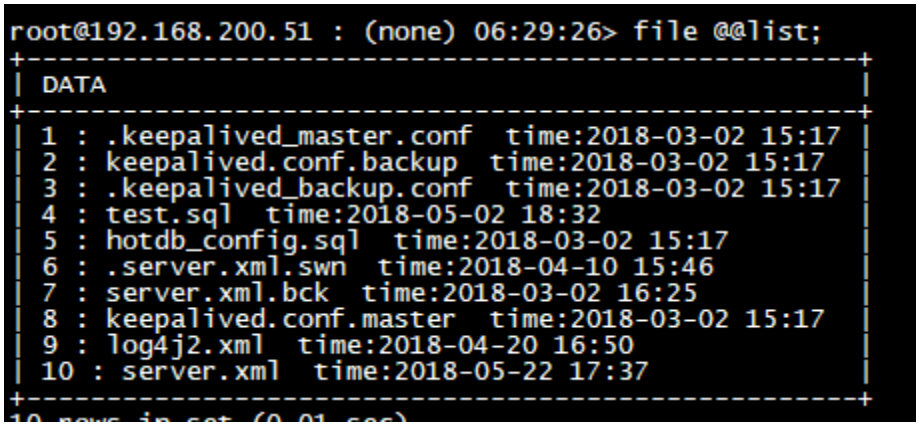
```
mysql> onlineddl "alter table mytb add column cl1 varchar(90) default '1'";
```

Note: when online modifying table structure, the data table structures on each sharding shall be consistent, and the data table to be modified has unique index.

5.4. file @@list – Obtain the files under the conf directory and its final modification time

This command is used to view and obtain the files under the conf directory and its final modification time. For example:

```
mysql> file @@list;
```



Fields and their description are contained in the result:

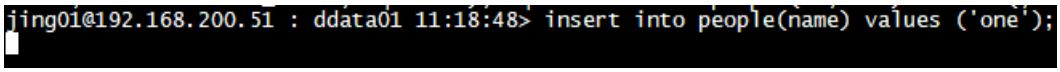
Column Name	Description	Value Type/Range
DATA	information of related files	STRING/[number : file

	under the conf directory	“time”:yyyy-MM-dd hh:mm:ss]

5.5. hold commit – Set connection status of all clients as HOLD_ALL_COMMIT

HOLD COMMIT is executed in the monitoring window of the command line of HotDB Server, and the commit of transaction in server port will be HOLD (including the transaction commit and normal autocommit). For example, when autocommitting the transaction type:

```
mysql> hold commit;

Query OK, 1 row affected (0.02 sec)
```



5.6. hold ddl – Set connection status of all clients as HOLD_DDL

HOLD DDL is executed in the monitoring window of the command line of HotDB Server, and the execution of related DDL statement in server port will be temporarily HOLD. For example:

```
mysql> hold ddl;

Query OK, 1 row affected (0.02 sec)
```

```
jing01@192.168.200.51 : ddata01 11:23:15> create table people2(id int primary key auto_increment,name varchar(19));
```

5.7. releasehold commit – Release the connection status of HOLD_ALL_COMMIT

After execution of hold commit, release HOLD status with this command, and transaction is committed successfully. For example:

```
mysql> releasehold commit;

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

```
jing01@192.168.200.51 : ddata01 11:18:48> insert into people(name) values ('one');
Query OK, 1 row affected (47.07 sec)
```

5.8. releasehold ddl – Set HOLD_DDL connection status as UNHOLD

After execution of hold ddl, release HOLD status with this command, and statement is executed successfully. For example:

```
mysql> releasehold ddl;

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

```
jing01@192.168.200.51 : ddata01 11:23:15> create table people2(id int primary key auto_increment,name varchar(19));
Query OK, 0 rows affected (28.63 sec)
```

5.9. Global unique constraint

For global unique constraint related contents, please refer to Distributed Transaction Database HotDB Server [Standard] Function Manual

5.9.1. check @@history_unique – Check the uniqueness of historical data of unique key

This command is used to check whether the historical data of unique constraint key of the specified table is unique. The statement is:

```
mysql> check @@history_unique [db_name.tb_name];
```

1. Table name is not specified: check all tables with global unique constraint, whether the historical data of their unique constrain key is unique. If it is unique, an empty set will be returned:

```
mysql> check @@history_unique;

Empty set (0.01 sec)
```

If there is a small amount of inconsistent data, inconsistent values will be prompted:

```
mysql> check @@history_unique;

+---------+---------+-----------------------------------------------+

| db_name | tb_name | messege                                       |

+---------+---------+-----------------------------------------------+

| DB1 | test1     | duplicate data in unique constraint: ID1:[2] |

+---------+---------+-----------------------------------------------+
```

If there is a large number of inconsistent data, with more than 2048 characters in length, you will be prompted to download files for check:

```
mysql> check @@history_unique;
+---------+---------+-----------------------------------------------------------------------------------------------------------------------------+
| ZJJ_DB1 | UCON1   | duplicate data in unique constraint, for more information,please download: ZJJ_DB1_UCON1_duplicates_1561353006576 |
+---------+---------+-----------------------------------------------------------------------------------------------------------------------------+
```

2. Table name is specified: check whether the historical data of unique constraint key of the specified table is unique. For example:

```
check @@history_unique db01.table01,db01.table02,db01.table03
```

5.9.2. unique @@create – create secondary index

This command is used to create the secondary index after the historical data of unique constraint key of the specified table is checked unique. The statement is:

```
mysql> unique @@create [db_name.tb_name];
```

1. Table name is not specified: check whether the unique constraint key of all tables is unique. If it is unique, the secondary index is created. For example:

```
mysql> unique @@create;
+------------------+-------------+---------+----------------------------+
| db_name          | tb_name     | result  | messege                    |
+------------------+-------------+---------+----------------------------+
| HOTDB_SERVER_253 | ORDERFORM   |    fail | global_unique is turned off |
| HOTDB_SERVER_253 | CLIENT      | success |                            |
| HOTDB_SERVER_253 | KEEVEY01    | success |                            |
```

```
+-----------------+------------+---------+--------------------------+
```

- If the secondary index is successfully created, then the result is success;
- If the global unique constraint of this table is in off state, then the result is fail, and the information is shown: global_unique is turned off;
- If the historical data is unique, but the secondary index fails to be created, then the result is fail, and the information error is shown;
- If the historical data is not unique, then the result is fail, and inconsistent result and the command check @@history_unique are shown.

2. Table name is specified: check whether the historical data of unique constraint key of the specified table is unique. For example:

```
unique @@create db01.table01,db01.table02,db01.table03
```

If this command contains the table whose secondary index has been created, the existing secondary index will be deleted and a new one will be created after the command is executed.

5.9.3. unique @@drop – Delete secondary index

This command is used to delete the secondary index of the specified table. The statement is:

```
mysql> unique @@drop [db_name.tb_name];
```

For example:

```
mysql> unique @@drop HOTDB_SERVER_253.beyond1,HOTDB_SERVER_253.test1,HOTDB_SERVER_25
3.keevey01;

+-----------------+----------+---------+---------+

| db_name | tb_name | result | messege |
```

```
+-----------------+----------+---------+---------+
| HOTDB_SERVER_253 | BEYOND1   | success |         |
| HOTDB_SERVER_253 | KEEVEY01 | success |         |
| HOTDB_SERVER_253 | TEST1     | success |         |
+-----------------+----------+---------+---------+
```

- If the secondary index is deleted successfully, then the result is success;
- If the secondary index fails to be deleted, then the result is fail and the error information is shown.