w: www.mariadb.com

e: info@mariadb.com

Limitations and Known Issues within MariaDB MaxScale

The purpose of this documentation is to provide a central location that will document known issues and limitations within the MaxScale product and the plugins that form part of that product. Since limitations may be related to specific plugins or to MaxScale as a whole this document is divided into a number of sections, the purpose of which are to isolate the limitations to the components which illustrate them.

Limitations in the MaxScale core

This section describes the limitations that are common to all configuration of plugins with MaxScale.

Limitations with MySQL Protocol support

- Compression
- SSL
  Both capabilities are not included in MySQL server handshake
- LOAD DATA LOCAL INFILE currently not supported

Limitations with MySQL Master/Slave Replication monitoring

Limitations with Galera Cluster Monitoring

Master selection is based only on MIN(wsrep_local_index), no other server parameter.

Limitations in the connection router

If Master changes (ie. new Master promotion) during current connection the router cannot check the change

Limitations in the Read/Write Splitter

Scale-out limitations

In master-slave replication cluster also read-only queries are routed to master too in the following situations:

- if they are executed inside an open transaction

- in case of prepared statement execution
- statement includes a stored procedure, or an UDF call

Limitations in client session handling

Some of the queries that client sends are routed to all backends instead of sending them just to one of server. These queries include “USE <db name>” and “SET autocommit=0” among many others. Read/Write Splitter sends a copy of these queries to each backend server and forwards the first reply it receives to the client. Below is a list of MySQL commands which we call session commands :

COM_INIT_DB (USE <db name> creates this)
COM_CHANGE_USER
COM_STMT_CLOSE
COM_STMT_SEND_LONG_DATA
COM_STMT_RESET
COM_STMT_PREPARE

Also these are session commands:

COM_QUIT (no response)
COM_REFRESH
COM_DEBUG
COM_PING

In addition there are query types which belong to the same group:

SQLCOM_CHANGE_DB
SQLCOM_DEALLOCATE_PREPARE
SQLCOM_PREPARE
SQLCOM_SET_OPTION
SELECT ..INTO variable|OUTFILE|DUMPFILE

Then there are queries which modify session characteristics, listed as derived, internal RWSplit types:

QUERY_TYPE_ENABLE_AUTOCOMMIT
QUERY_TYPE_DISABLE_AUTOCOMMIT

There is a possibility for misbehavior; if “USE mytable” was executed in one of the slaves and it failed, it may be due to replication lag rather than the fact it didn’t exist. Thus the same command may end up with different result among backend servers. This disparity is missed.

The above-mentioned behavior can be partially controller with RWSplit configuration parameter called

```
use_sql_variables_in=[master|all] (master)
```

Server-side session variables are called as SQL variables. If “master” or no value is set, SQL variables are read and written in master only. Autocommit values and prepared statements are routed to all nodes always.

NOTE: If variable is written as a part of write query, it is treated like write query and not routed to all servers. For example, `INSERT INTO test.t1 VALUES (@myvar:= 7)`.

Examples:
If new database “db” was created and client executes “USE db” and it is routed to slave before the CREATE DATABASE clause is replicated to all slaves there is a risk of executing query in wrong database. Similarly, if any response that RWSplit sends back to the client differ from that of the master, there is a risk for misbehavior.

Most imaginable reasons are related to replication lag but it could be possible that a slave fails to execute something because of some non-fatal, temporary failure while execution of same command succeeds in other backends.

Authentication Related Limitations

MySQL old passwords are not supported