

Differences between MariaDB and MySQL — MariaDB 10.3 vs. MySQL 8.0

Colin Charles, Chief Evangelist, Percona Inc.
colin.charles@percona.com / byte@bytebot.net
<http://bytebot.net/blog/> | @bytebot on Twitter
Percona Webinar
18 July 2018



whoami



- Chief Evangelist, Percona Inc
- Founding team of MariaDB Server (2009-2016) [Monty Program Ab, merged with SkySQL Ab, now MariaDB Corporation]
- Formerly MySQL AB (exit: Sun Microsystems)
- Past lives include The Fedora Project (FESCO), OpenOffice.org
- MySQL Community Contributor of the Year Award winner 2014

License

- Creative Commons BY-NC-SA 4.0
- <https://creativecommons.org/licenses/by-nc-sa/4.0/legalcode>



A good base blog post resource

- High level, *answer* to a whitepaper
- <https://www.percona.com/blog/2017/11/02/mysql-vs-mariadb-reality-check/>
- Watch the blog for updates after this webinar (since we do want to compare MySQL 8 to MariaDB Server 10.3)
 - and also, watch it after we release Percona Server for MySQL 8!
- Versions compared in this webinar:
 - MySQL 8.0.11 (released: 19 April 2018)
 - MariaDB Server 10.3.8 (released: 2 July 2018)

Define: compatibility (OED)

- A state in which two things are able to exist or occur together without problems or conflict.

1.2 *Computing* The ability of one computer, piece of software, etc. to work with another.

'software compatibility is another important factor to consider'

Why this matters

- MariaDB Server is the “**default**” MySQL in many Linux distributions
 - not Ubuntu! (shipping MySQL 5.7)
- MariaDB isn't MySQL - there are many cloud providers that have an option for MariaDB
 - Amazon Web Services (AWS) RDS MariaDB, Microsoft Azure, Rackspace Cloud
- There are incompatibilities (e.g. with connectors)
 - <https://github.com/brianmario/mysql2/issues/878>
 - MariaDB Connector/C for MySQL and MariaDB Server. It is libmysqlclient API compatible. LGPL. OpenSSL/GnuTLS/schannel (no more yaSSL/wolfSSL)
 - header change in MariaDB 10.2.6 and mysqlclient python binding - <https://lists.launchpad.net/maria-developers/msg10726.html>

Commitments

- Verbal commitments: **“MySQL 5.6, should be comparable to MariaDB Server 10.1. And for 10.2 it should be compatible with MySQL 5.7”** — Michael “Monty” Widenius, CTO of MariaDB Corporation and MariaDB Foundation, 7 October 2016, MariaDB Developer’s Meeting, Amsterdam
- <http://mariadb.org/about/>
- “It is an enhanced, drop-in replacement for MySQL.”

Licensing

- MariaDB Server: GPLv2 only
- MariaDB MaxScale: Business Source License
- MariaDB ColumnStore: GPLv2 only
 - Backup & Restore, ColumnStore Kafka data adapter, MariaDB MaxScale CDC Data Adapter: Business Source License
- MySQL has Community (GPLv2) and Enterprise releases

Support










- What is the support ecosystem and landscape like?
- Training?
- MySQL Certification (MariaDB Corporation started certification at MySQL 17)

Community Contributions

- Oracle Contributor Agreement (OCA)
- MariaDB Contributor Agreement (MCA)
- BSD New
- Who maintains the code? What is the state of community contributed code?

Governance

- MariaDB Corporation
- MariaDB Foundation
- Is there vendor lock-in in open source?
- How many users are there, really?

Number of Funding Rounds		9	Total Funding Amount		\$98.2M
Announced Date	Transaction Name	Number of Investors	Money Raised	Lead Investors	
Sep 29, 2017	 Series C - MariaDB	6	\$27M	Alibaba	
May 8, 2017	 Venture Round - Mar..	1	€25M	European Investment Bank..	
Apr 5, 2016	 Series B - MariaDB	1	\$3M	–	
Jan 21, 2016	 Series B - MariaDB	2	\$9M	–	
Feb 26, 2015	 Series B - MariaDB	1	€3M	–	
Oct 23, 2013	 Series B - MariaDB	5	\$20M	Intel Capital	
May 30, 2012	 Series A - MariaDB	1	\$2.5M	–	
Apr 4, 2012	 Series A - MariaDB	3	\$4M	OnCorps, Inc.	
Sep 1, 2010	 Venture Round - Mar..	1	\$2M	–	

Releases

MariaDB	MySQL
5.1: 1 Feb 2010	5.1: 14 Nov 2008
5.2: 10 Nov 2010	
5.3: 29 Feb 2012	
5.5: 11 Apr 2012	5.5: 3 Dec 2010
	5.6: 5 Feb 2013
10.0: 31 Mar 2014	
10.1: 17 Oct 2015	
	5.7: 21 Oct 2015
10.2: 23 May 2017	
	8.0: 19 Apr 2018
10.3: 25 May 2018	



Shane K Johnson

@shane_dev

Follow



MariaDB left MySQL behind a while ago.

3:56 PM - 30 Jun 2018

1 Retweet 1 Like



1



1



Tweet your reply



Shane K Johnson

@shane_dev

#NoSQL, #BigData, #IoT, and anything distributed. Powered by beer + SciFi. Learned open source from the monks at Red Hat. #MariaDB - The O.G. is back.

What's documented?

- <https://mariadb.com/kb/en/the-mariadb-library/mariadb-vs-mysql-compatibility/>
- (closed!?) Tracker bug: <https://jira.mariadb.org/browse/MDEV-10392>
- <https://mariadb.com/kb/en/library/system-variable-differences-between-mariadb-and-mysql/>
- <https://mariadb.com/kb/en/library/incompatibilities-and-feature-differences-between-mariadb-102-and-mysql-57/>

Replication

Replication Compatibility

Slave↓ Master→	MariaDB- 5.5	MariaDB- 10.0	MariaDB- 10.1	MariaDB- 10.2	MySQL- 5.6	MySQL- 5.7	MySQL-8.0
MariaDB-5.5	Ok	No	No	No	No	No	No
MariaDB-10.0	Ok	Ok			Ok		
MariaDB-10.1	Ok	Ok	Ok		Ok		
MariaDB-10.2	Ok	Ok	Ok	Ok	Ok	Ok	
MySQL-5.6					X	X	X
MySQL-5.7					X	X	X
MySQL-8.0					X	X	X

Note X: Refer to MySQL documentation

Note: When replication from MySQL in GTID mode, MariaDB will remove the MySQL GTID events and replace them with MariaDB GTID events.

GTID variances between MariaDB & MySQL

- <https://mariadb.com/kb/en/library/gtid/#the-domain-id>

```
[MariaDB [(none)]> show global variables like '%gtid%';
```

Variable_name	Value
gtid_binlog_pos	0-1-9
gtid_binlog_state	0-1-9
gtid_current_pos	0-1-9
gtid_domain_id	0
gtid_ignore_duplicates	OFF
gtid_slave_pos	
gtid_strict_mode	OFF
wsrep_gtid_domain_id	0
wsrep_gtid_mode	OFF

9 rows in set (0.00 sec)

```
mysql> show global variables like '%gtid%';
```

Variable_name	Value
binlog_gtid_simple_recovery	ON
enforce_gtid_consistency	OFF
gtid_executed	
gtid_executed_compression_period	1000
gtid_mode	OFF
gtid_owned	
gtid_purged	
session_track_gtid	OFF

8 rows in set (0.01 sec)

Replication

- Default binlog format is now MIXED (ROW in MySQL)
- Default `replicate_annotate_row_events` is ON
- Binlog event compression - `log_bin_compress`
- Time delayed replication (present in MySQL 5.6; arrived in MariaDB 10.2)
- `read_binlog_speed_limit` - restricting the speed at which the slave reads the binlog from the master
- DML only Flashback - rollback instances/databases/tables to an older snapshot (via Alibaba!)
- Continuous streaming binary log backup added to `mysqlbinlog` in 10.2

5.1/5.2

- `mysqld` reads `[mariadb]` part of `my.cnf` for MariaDB Server only options
- Binary-only storage engines won't work without recompilation due do different THD structure (e.g. commercial engines like ScaleDB)
- Extended slow query log statistics (microslow patch from Percona)
- More memory utilised: Aria used to handle internal temporary tables, needs configuration
- MariaDB only: table elimination

5.3

- Error numbers for MariaDB are at 1900+; MySQL has to deal: <https://bugs.mysql.com/bug.php?id=72062>
- Microseconds arrived; but got *fixed* in MariaDB 10.1 to follow the MySQL 5.6 format
- SHOW PROCESSLIST with progress reporting
- New features: dynamic columns, virtual columns (5.7), HandlerSocket plugin, Cassandra storage engine (now deprecated)
- Huge changes in optimiser + replication
- <https://mariadb.com/kb/en/library/optimizer-feature-comparison-matrix/>

```
MariaDB [(none)]> select * from INFORMATION_SCHEMA.processlist\G
***** 1. row *****
      ID: 13
     USER: root
     HOST: localhost
        DB: NULL
  COMMAND: Query
         TIME: 0
        STATE: Filling schema table
        INFO: select * from INFORMATION_SCHEMA.processlist
     TIME_MS: 0.468
        STAGE: 0
     MAX_STAGE: 0
     PROGRESS: 0.000
  MEMORY_USED: 85096
 EXAMINED_ROWS: 0
     QUERY_ID: 28
  INFO_BINARY: select * from INFORMATION_SCHEMA.processlist
         TID: 19574
```

```
      Info: NULL
     Progress: 0.000
***** 6. row *****
      Id: 13
     User: root
     Host: localhost
        db: NULL
  Command: Query
         Time: 0
        State: init
        Info: show processlist
     Progress: 0.000
  6 rows in set (0.00 sec)
```

```
[mysql> select * from INFORMATION_SCHEMA.processlist\G
***** 1. row *****
      ID: 39296
     USER: root
     HOST: localhost
        DB: NULL
  COMMAND: Query
         TIME: 0
        STATE: executing
        INFO: select * from INFORMATION_SCHEMA.processlist
     TIME_MS: 0
     ROWS_SENT: 0
  ROWS_EXAMINED: 0
  1 row in set (0.00 sec)
```

```
[mysql> show processlist\G
***** 1. row *****
      Id: 39296
     User: root
     Host: localhost
        db: NULL
  Command: Query
         Time: 0
        State: starting
        Info: show processlist
     Rows_sent: 0
  Rows_examined: 0
  1 row in set (0.00 sec)
```

10.0

Incompatibilities between MariaDB 10.0 and MySQL 5.6

- MySQL does not support MariaDB's [Spider Storage Engine](#).
- All MySQL binaries (`mysqld` , `myisamchk` etc.) give a warning if one uses a prefix of an option (such as `--big-table` instead of `--big-tables`). MariaDB binaries work in the same way as most other Unix commands and don't give warnings when using unique prefixes.
- MariaDB GTID is not compatible with MySQL 5.6. This means that one can't have MySQL 5.6 as a slave for [MariaDB 10.0](#). However [MariaDB 10.0](#) can be a slave of MySQL 5.6 or any earlier MySQL/MariaDB version. Note that MariaDB and MySQL also have different [GTID system variables](#), so these need to be adjusted when migrating.
- [MariaDB 10.0 multi-source replication](#) is not supported in MySQL 5.6.
- To make `CREATE TABLE ... SELECT` work the same way in statement based and row based replication it's by default executed as [CREATE OR REPLACE TABLE](#) on the slave. One benefit of this is that if the slave dies in the middle of `CREATE ... SELECT` it will be able to continue.
 - One can use the [slave-ddl-exec-mode](#) variable to specify how `CREATE TABLE` and `DROP TABLE` is replicated.
- See also a detailed breakdown of [System variable differences between MariaDB 10.0 and MySQL 5.6](#).
- MySQL 5.6 has [performance schema](#) enabled by default. For performance reasons [MariaDB 10.0](#) has it disabled by default. You can enable it by starting `mysqld` with the option `--performance-schema` .
- [MariaDB 10.0](#) does not support the MySQL Memcached plugin. However, data stored using memcached can be retrieved because the data is stored as InnoDB tables. MariaDB is able to start successfully with an error message of not being able to find `libmemcached.so` library.
- Users created with MySQL's SHA256 password algorithm cannot be used in [MariaDB 10.0](#) as MariaDB does not include MySQL's `sha256_password` plugin.
- [MariaDB 10.0](#) does not support delayed replication - [MDEV-7145](#).
- Also see a detailed breakdown of [System variable differences between MariaDB 10.0 and MySQL 5.6](#).
- The low-level temporal format used by `TIME`, `DATETIME` and `TIMESTAMP` is different in MySQL 5.6 and [MariaDB 10.0](#). (In [MariaDB 10.1](#), the MySQL implementation is used by default - see [mysql56_temporal_format](#).)
- MariaDB implements some changes in the SQL query optimizer over what's available in MySQL. This can result in [EXPLAIN](#) statements showing different plans.
- MySQL delayed replication, (through `MASTER_DELAY`), is not supported in [MariaDB 10.0](#), it was implemented in [MariaDB 10.2.5](#)

10.1

Incompatibilities between MariaDB 10.1 and MySQL 5.7

- [MariaDB 10.1](#) and above does not support MySQL 5.7's packed JSON objects. MariaDB follows the SQL standard and stores the JSON as a normal TEXT/BLOB. If you want to replicate JSON columns from MySQL to MariaDB, you should store JSON objects in MySQL in a TEXT column or use statement based replication. If you are using JSON columns and want to upgrade to MariaDB, you can either convert the JSON columns to TEXT or use [mysqldump](#) to copy these tables to MariaDB. In MySQL, JSON is compared according to json values. In MariaDB JSON strings are normal strings and compared as strings.
- [MariaDB 10.1](#)'s InnoDB encryption is implemented differently than MySQL 5.7's InnoDB encryption.
- [MariaDB 10.1](#) does not support the ngram and MeCab full-text parser plugins - [MDEV-10267](#), [MDEV-10268](#).
- [MariaDB 10.1](#) does not support multiple triggers for a table - [MDEV-6112](#). This is fixed in [MariaDB 10.2](#)
- [MariaDB 10.1](#) does not support [CREATE TABLESPACE](#) for InnoDB.
- [MariaDB 10.1](#) does not support MySQL 5.7's "native" InnoDB partitioning handler.
- MariaDB does not support MySQL 5.7's X protocol.
- [MariaDB 10.1](#) does not support the use of multiple triggers of the same type for a table. This feature was introduced in [MariaDB 10.2.2](#).
- [MariaDB 10.1](#) does not support MySQL 5.7's transportable tablespaces for partitioned InnoDB tables. ALTER TABLE ... {DISCARD|IMPORT} PARTITION is not supported. For a [workaround see the following blog post](#).
- [MariaDB 10.1](#) does not support MySQL 5.7's online undo tablespace truncation. However, this feature was added to [MariaDB 10.2](#).
- MySQL 5.7 features a new implementation of the `performance_schema` and a `sys` schema wrapper. These are not yet supported in MariaDB.
- MySQL 5.7 adds multi-source replication and replication channels. [Multi-source replication](#) was added to MariaDB previously, in [MariaDB 10.0](#), and uses a different syntax.
- MySQL 5.7 adds group replication. This feature is incompatible with MariaDB's [galera-cluster](#) replication.
- [MariaDB 10.1](#) does not support MySQL 5.7's, ACCOUNT LOCK/UNLOCK syntax for CREATE USER and ALTER USER statements.
- [MariaDB 10.1](#) does not support MySQL 5.7's ALTER TABLE...RENAME INDEX statements.
- [MariaDB 10.1](#) does not support MySQL 5.7's STACKED operation for [GET DIAGNOSTICS](#) statements.
- [MariaDB 10.1](#) does not support MySQL 5.7's {WITH|WITHOUT} VALIDATION syntax for ALTER TABLE... EXCHANGE PARTITION statements.
- Also see [Incompatibilities between MariaDB 10.0 and MySQL 5.6](#).
- Also see a [detailed breakdown of System variable differences between MariaDB 10.1 and MySQL 5.7](#).

10.2

Incompatibilities and Feature Differences Between MariaDB 10.2 and MySQL 5.7

[Home](#)

[Open Questions](#)

[MariaDB](#)

[MariaDB MaxScale](#)

[MariaDB ColumnStore](#)

[Connectors](#)

[All Topics](#)

[History](#)

[Source](#)

[Flag as Spam / Inappropriate](#)

[Translate](#)

Created
10 months, 4 weeks ago
Modified
8 days, 1 hour ago
Type
article
Status
active
License
CC BY-SA / Gnu FDL

[History](#)
[Comments](#)

Links

- [MySQL X plugin](#)
- [here](#)
- [more than 2x times faster](#)
- [replication notably faster!](#)

The following is a list of incompatibilities and feature differences between [MariaDB 10.2](#) and [MySQL 5.7](#).

Storage Engines

In addition to the standard [MyISAM](#), [BLACKHOLE](#), [CSV](#), [MEMORY](#), [ARCHIVE](#), and [MERGE](#) storage engines, the following are also available with [MariaDB 10.2](#):

- [MyRocks](#), a storage engine with great compression
- [Aria](#), [MyISAM](#) replacement with better caching.
- [TokuDB](#)
- [CONNECT](#)
- [SEQUENCE](#)
- [SphinxSE](#)
- [Spider](#)
- [FederatedX](#) (drop-in replacement for [Federated](#))
- [OOGRAPH](#)
- [Cassandra](#)

Speed Improvements

- Many optimizer enhancements. [Subqueries](#) are more usable. The complete list and a comparison with MySQL is [here](#). A benchmark can be found [here](#).
- Faster and safer replication: [Group commit for the binary log](#). This makes many setups that use replication and lots of updates [more than 2x times faster](#).
- [Improvements](#) for [InnoDB](#) asynchronous IO subsystem on Windows.
- [Segmented Key Cache](#) for [MyISAM](#). Can speed up [MyISAM](#) tables with up to 4x
- [Adjustable hash size](#) for [MyISAM](#) and [Aria](#). This can greatly improve shutdown time (from hours to minutes) if you are using a lot of [MyISAM/Aria](#) tables with delayed keys.
- [CHECKSUM TABLE](#) is faster.
- We improved the performance of character set conversions (and removed conversions when they were not really needed). Overall speed improvement is 1-5 % (according to [sql-bench](#)) but can be higher for big result sets with all characters between 0x00-0x7F.
- [MariaDB Thread pool](#) allows MariaDB to run with 200,000+ connections and with a notable speed improvement when using many connections.
- Lots of speed improvements when a client connects to MariaDB.
- There are some improvements to the [DEBUG](#) code to make its execution faster when debug is compiled in but not used.
- Our use of the [Aria](#) storage engine enables faster complex queries (queries which normally use disk-based temporary tables). The [Aria](#) storage engine is used for internal temporary tables, which should give a speedup when doing complex selects. [Aria](#) is usually faster for temporary tables when compared to [MyISAM](#) because [Aria](#) caches row data in memory and normally doesn't have to write the temporary rows to disk.
- The test suite has been extended and faster than before, even though it tests more things.

Contents

1. [Storage Engines](#)
2. [Speed Improvements](#)
3. [Extensions and New Features](#)
4. [Incompatibilities](#)

1. [Compatibility & Differences 1](#)

[MariaDB versus MySQL - Compatibility](#)

[MariaDB versus MySQL - Features](#)

[Incompatibilities and Feature Differences Between MariaDB 10.2 and MySQL 5.7](#)

[Optimizer Feature Comparison Maria](#)

[Differences Between the MySQL and MariaDB Query Optimizer](#)

[System Variable Differences between MariaDB and MySQL](#)

[Function Differences between MariaDB and MySQL](#)

[SQL_MODE=ORACLE From MariaDB 10.3](#)

Extensions and New Features

We've added a lot of [new features to MariaDB](#). If a patch or feature is useful, safe, and stable — we make every effort to include it in MariaDB. The most notable features are:

10.2

- [Window functions](#)
- Number of supported decimals in [DECIMAL](#) has increased from 36 to 38
- [Recursive Common Table Expressions](#)
- New [WITH](#) statement. [WITH](#) is a common table expression that allows you to refer to a subquery expression many times in a query.
- [CHECK CONSTRAINT](#)
- [DEFAULT](#) expression, including [DEFAULT](#) for [BLOB](#) and [TEXT](#)
- Added catchall for [list partitions](#)
- Oracle-style [EXECUTE IMMEDIATE](#) statement
- Lots of new [JSON](#) functions
- [Microsecond Precision in Processlist](#)
- [Table Elimination](#)
- [Virtual Columns](#)
- [Extended User Statistics](#)
- [KILL](#) all queries for a user
- [KILL QUERY ID](#) - terminates the query by `query_id`, leaving the connection intact
- [Pluggable Authentication](#)
- [Storage-engine-specific CREATE TABLE](#)
- Enhancements to [INFORMATION_SCHEMA.PLUGINS](#) table
- [Group commit for the binary log](#). This makes [replication](#) notably faster!
- Added `--newrite-db=mysqlbinlog` option to change the used database
- [Progress reporting for ALTER TABLE and LOAD DATA INFILE](#)
- [Faster joins and subqueries](#)
- [HandlerSocket](#) and faster [HANDLER](#) calls
- [Dynamic Columns](#) support
- [SHOW EXPLAIN](#) gives the [EXPLAIN](#) plan for a query running in another thread.
- [Roles](#)
- [PCRE Regular Expressions](#) (including [REGEXP_REPLACE\(\)](#))
- [DELETE ... RETURNING](#)
- MariaDB supports expressions in the [DEFAULT](#) clause, while MySQL does not.
- MariaDB supports more collations than MySQL, including [NO PAD](#) collations.

10.2

Incompatibilities

When upgrading from MySQL 5.7 to MariaDB 10.2, please take note of the following incompatibilities:

- For a list of function differences, see [Function Differences Between MariaDB 10.2 and MySQL 5.7](#).
- For a list of system variable differences, see [System Variable Differences Between MariaDB 10.2 and MySQL 5.7](#).
- MySQL binaries (`mysqld`, `mysqldchk` etc.) give a warning if one uses a prefix of an option (such as `--big-table` instead of `--big-tables`). MariaDB binaries work in the same way as most other Unix commands and don't give warnings when using unique prefixes.
- MariaDB's GTID is not compatible with MySQL's. This means that one can't have MySQL 5.7 as a slave for MariaDB 10.2. However MariaDB 10.2 can be a slave of MySQL 5.7 or any earlier MySQL/MariaDB version. Note that MariaDB and MySQL also have different [GTID system variables](#), so these need to be adjusted when migrating.
- To make `CREATE TABLE ... SELECT` work the same way in statement based and row based replication it's by default executed as `CREATE OR REPLACE TABLE` on the slave. One benefit of this is that if the slave dies in the middle of `CREATE ... SELECT` it will be able to continue.
 - One can use the `slave-ddl-exec-mode` variable to specify how `CREATE TABLE` and `DROP TABLE` is replicated.
- MySQL has the [performance schema](#) enabled by default. For performance reasons MariaDB 10.2 has it disabled by default. You can enable it by starting `mysqld` with the option `--performance-schema`.
- MariaDB 10.2 implements [InnoDB encryption](#) in a different way to MySQL 5.7.
- MariaDB 10.2 does not support `CREATE TABLESPACE` for InnoDB.
- The `OVER`, `ROWS` and `RECURSIVE` keywords are [reserved words](#) in MariaDB 10.2, but not in MySQL 5.7. Note that in MySQL 5.0 these are also reserved words.
- MariaDB stores [JSON](#) as true text, not in binary format as MySQL. MariaDB's JSON functions are much faster than MySQL's so there is no need to store in binary format, which would add complexity when manipulating JSON objects.
- For the same reason, MariaDB's [JSON data type](#) is an alias for `LONGTEXT`. If you want to replicate JSON columns from MySQL to MariaDB, you should store JSON objects in MySQL in a `TEXT` or `LONGTEXT` column or use statement based replication. If you are using JSON columns and want to upgrade to MariaDB, you need to either convert them to `TEXT` or use `mysqldump` to copy these tables to MariaDB.
- In MySQL, JSON is compared according to json values. In MariaDB JSON strings are normal strings and compared as strings.
- MariaDB 10.2 does not support MySQL's JSON operators (`->` and `->>`).
- MariaDB 10.2 supports the standard by producing null and a warning for `JSON_SEARCH` when given invalid data, while MySQL produces an error.
- MariaDB 10.2 does not support the ngram and McCab full-text parser plugins - [MDEV-10267](#), [MDEV-10268](#).
- MariaDB 10.2 does not support the [MySQL X plugin](#).
- MariaDB 10.2 does not support MySQL 5.7's "naive" InnoDB partitioning handler.
- MySQL's implementation of [aborting statements that exceed a certain time to execute](#) can only kill SELECTs, while MariaDB's can kill any queries (excluding stored procedures).
- MariaDB 10.2 does not support MySQL's `SELECT MAX_STATEMENT_TIME = N ...` syntax - see [Aborting Statements that Exceed a Certain Time to Execute](#).
- The MySQL version of `max_statement_time` is defined in milliseconds, not seconds.
- MariaDB 10.2 does not support the MySQL `memcached` plugin. However, data stored using `memcached` can be retrieved because the data is stored as InnoDB tables. MariaDB is able to start successfully with an error message of not being able to find `libmemcached.so` library.
- Users created with MySQL's SHA256 password algorithm cannot be used in MariaDB 10.2.
- In MySQL, `X'HHHH'`, the standard SQL syntax for binary string literals, erroneously works in the same way as `BBBB`, which could work as a number or string depending on the context. In MariaDB, this has been fixed to behave as a string in all contexts (and never as a number). See [CONST](#) and [Hexadecimal Literals](#) for more details and examples.
- In MariaDB 10.2, `SHOW CREATE TABLE` does not quote the `DEFAULT` value of an integer. Older versions of MariaDB and MySQL, do. Since MariaDB 10.2 can support defaults for `BLOB` and `TEXT` fields, while MySQL does not, `SHOW CREATE TABLE` will also append `DEFAULT NULL` where no default is explicitly provided to nullable `BLOB` or `TEXT` fields in MariaDB.
- Since MariaDB supports expressions in the `DEFAULT` clause, in MariaDB, the `INFORMATION_SCHEMA.COLUMNS` table contains extra fields, and also quotes the `DEFAULT` value of a string in the `COLUMN_DEFAULT` field in order to distinguish it from an expression.
- Also see [Incompatibilities between MariaDB 10.1 and MySQL 5.7](#).

Configuring mysql-community-server

MySQL 8 uses a new authentication based on improved SHA256-based password methods. It is recommended that all new MySQL Server installations use this method going forward. This new authentication plugin requires new versions of connectors and clients, with support for this new 8 authentication (caching_sha2_password). Currently MySQL 8 Connectors and community drivers built with libmysqlclient21 support this new method. Clients built with older versions of libmysqlclient may not be able to connect to the new server.

To retain compatibility with older client software, the default authentication plugin can be set to the legacy value (`mysql_native_password`). This should only be done if required third-party software has not been updated to work with the new authentication method. The change will be written to the file `/etc/mysql/mysql.conf.d/default-auth-override.cnf`.

After installation, the default can be changed by setting the `default_authentication_plugin` server setting.

Select default authentication plugin

Use Strong Password Encryption (RECOMMENDED)

Use Legacy Authentication Method (Retain MySQL 5.x Compatibility)

<Ok>

JSON

- 5.7 has a binary data type, MariaDB chooses not to implement it this way, choosing to not “violate the SQL standard” - <https://jira.mariadb.org/browse/MDEV-9144>
- claims it is as fast, but there are no benchmarks - <https://jira.mariadb.org/browse/MDEV-13777>

X Protocol

- MariaDB Server has **no** support for the MySQL X Protocol
- This means you cannot use `mysqlsh` to access MariaDB Server
- This is *significant* as a limitation

Encryption

- MySQL 5.7 and MariaDB Server 10.1+ implement encryption differently (one is fully tablespace encryption, the other is based on the Google patch for tablespace encryption in addition to having table encryption via Eperi)
- MySQL 8 encrypts redo/undo logs via configuration options; temporary tablespace or binary log encryption does not exist (MariaDB supports binary log encryption, and temporary table encryption)
- MySQL requires `innodb_file_per_table`
- MySQL implementation works fully with Percona XtraDB Cluster
 - MariaDB Galera Cluster `gcache` is unencrypted - <https://jira.mariadb.org/browse/MDEV-9639>
- `mysqlbinlog` cannot read encrypted binary logs - <https://jira.mariadb.org/browse/MDEV-8813>
 - workaround is use `—read-from-remote-server` as the server has access to the keys via the encryption plugin API. This adds load to the server!
- Backups: you have to use MariaDB Backup for encrypted backups (Percona XtraBackup fork; upstream does not work)
- Key management: Percona has Vault, MariaDB has an AWS Key Management Plugin, MySQL has several but they are part of the Enterprise release

CJK language support

- Add “ngram” support to MariaDB Server: <https://jira.mariadb.org/browse/MDEV-10267>
- Add “MeCab” support to MariaDB Server: <https://jira.mariadb.org/browse/MDEV-10268>
- gb18030 support: <https://jira.mariadb.org/browse/MDEV-7495>

PERFORMANCE_SCHEMA

- No sys schema - <https://jira.mariadb.org/browse/MDEV-9077>
- No new PERFORMANCE_SCHEMA instrumentation from 5.7 - <https://jira.mariadb.org/browse/MDEV-6114>
- e.g. 52 rows in set (0.00 sec) vs. 87 rows in set (0.00 sec)

Security

- MySQL: `caching_sha256_password`
- MariaDB: `ed25519` password plugin
- Needless to say the above are **incompatible** with each other
- `validate_password` is on by default in MySQL 5.7 (not in MariaDB)

Other bits

- MariaDB: SHOW EXPLAIN FOR <thread_id>
- MySQL: EXPLAIN FOR CONNECTION <thread_id>
- <https://jira.mariadb.org/browse/MDEV-10000>
- MySQL has SUPER READONLY, missing in MariaDB - <https://jira.mariadb.org/browse/MDEV-9458>
- Optimiser trace: <https://jira.mariadb.org/browse/MDEV-6111>
- Replication crash-safety for non-GTID slaves - <https://jira.mariadb.org/browse/MDEV-8946>
- Minimal/NOBLOB Binlog Row Image replication fails when tables from master have different PK in slave - <https://jira.mariadb.org/browse/MDEV-8398>

Installation...

- MySQL: <https://dev.mysql.com/downloads/repo/yum/> - grab a package — e.g. `mysql57-community-release-el7-9.noarch.rpm`
- MariaDB Server: <https://downloads.mariadb.org/mariadb/repositories/> - copy/paste, edit a file, then install
- Percona Server: https://www.percona.com/doc/percona-server/5.7/installation/yum_repo.html - yum install `Percona-Server-server-57`

Starting up

- MySQL / Percona Server
 - `service mysqld start`
 - `grep 'temporary password' /var/log/mysqld.log`
 - `ALTER USER 'root'@'localhost' IDENTIFIED BY 'rootmeOK!';`
- MariaDB Server? `mysql -uroot` “just works”

mysql.user table changes

- MariaDB Server and MySQL differ here (not just by addition of roles)
 - `mysql.user.password` is just `mysql.user.authentication_string`
- Password expiry is coming? <https://jira.mariadb.org/browse/MDEV-7597>
- Password last changed? Lifetime?
- ACCOUNT LOCK/UNLOCK
- VALIDATE_PASSWORD_STRENGTH() SQL function doesn't work in MariaDB Server

```
max_questions: 10
max_updates: 20
max_connections: 30
max_user_connections: 1
plugin:
authentication_string:
password_expired: N
is_role: N
default_role:
max_statement_time: 0.000000
```

```
max_questions: 10
max_updates: 20
max_connections: 30
max_user_connections: 1
plugin: mysql_native_password
authentication_string:
password_expired: N
password_last_changed: 2017-05-05 04:58:04
password_lifetime: NULL
account_locked: N
```

More 5.7

- Optimizer hints (and the cost based optimizer itself?) - <https://jira.mariadb.org/browse/MDEV-9078>
- RENAME INDEX - <https://jira.mariadb.org/browse/MDEV-7318>
- Query rewriting? - <https://jira.mariadb.org/browse/MDEV-5561>
- GIS: GeoJSON functions? Geohash functions?
- `SELECT ST_AsGeoJSON(ST_GeomFromText('POINT(11.11111 12.22222)'),2);` [this works in MariaDB Server 10.3!]
- `SELECT ST_GeoHash(180,0,10), ST_GeoHash(-180,-90,15);` [still missing in MariaDB Server 10.3!]

Niggling *usability* bits

```
[mysql> SELECT ST_AsGeoJSON(ST_GeomFromText('POINT(11.1111 12.22222)'),2);  
[      '> ^C  
mysql> █
```

```
MariaDB [(none)]> SELECT ST_AsGeoJSON(ST_GeomFromText('POINT(11.1111 12.22222)'  
) ,2);  
      '> Ctrl-C -- exit!  
Aborted
```

- <https://jira.mariadb.org/browse/MDEV-14448>

Tools

- including new tools like `mysql_ssl_rsa_setup` ? `mysqlpump`?
- why will `xtrabackup` not work with MariaDB Server encryption or compression? <https://jira.mariadb.org/browse/MDEV-10367>
- answer: fork `xtrabackup` to call it MariaDB Backup
- When merging XtraDB, why isn't it complete, with backup locks? <https://jira.mariadb.org/browse/MDEV-5336>
- tools that require MySQL GTID don't work with MariaDB Server (e.g. `mysqlfailover`, `mysqlrpladmin`, MHA, MySQL Router, etc.)
- vitess, started life on just MariaDB Server, but note they support 5.6/5.7 and only 10.0

What is MariaDB TX 3.0?

- A combination of all the offerings MariaDB Corporation & MariaDB Foundation work on
- MariaDB Foundation: MariaDB Server (GPLv2), MariaDB Connectors for C/Java/ODBC (LGPL)
- MariaDB Corporation: services (remote DBA, migration, consulting, technical support) + MariaDB MaxScale proxy (Business Source License) + MariaDB Backup (fork of Percona XtraBackup), MariaDB Admin (SQLYog) + Monitor (Monyog) + Notifications (security alerts through a portal)
- TX Cluster includes support for Galera Cluster; AX for MariaDB ColumnStore

Key focus points for MariaDB 10.3

- Oracle compatibility
- More storage engines
- Temporal data (system versioned tables)
- Plus some of the features from 10.2+10.1+10.0+5.5+5.3+5.2+5.1 that *may* not be in stock MySQL

MariaDB storage engine offerings

- **MyRocks**: for write-intensive workloads
- **SPIDER**: for scalability and sharding
- **InnoDB**: default for read/write operations (no longer Percona XtraDB since MariaDB 10.2)
- **ColumnStore**: analytical purposes (not included in MariaDB Server 10.3 — still a separate download)
- **OQGRAPH**: leaves algorithm
 - note: requires libJudy
- **PARTITION**: updates to make SPIDER work better
- **Cassandra**: still around, requires libthrift
- **CONNECT**: for ETL operations
- **TokuDB**: requires jemalloc and transparent hugepages to be never (not always)

Storage Engines

- InnoDB 5.7 is now included in MariaDB Server 10.2 (there is no longer Percona XtraDB for the first time)
 - you need to remove XtraDB related options in my.cnf or the server won't start
 - <https://lists.launchpad.net/maria-discuss/msg04708.html>
- Is InnoDB fully tested?
 - Test cases still need merging - <https://jira.mariadb.org/browse/MDEV-13626>
 - Slow starts - <https://jira.mariadb.org/browse/MDEV-13869> / <https://lists.launchpad.net/maria-discuss/msg04922.html>
 - Hangs on startup - <https://jira.mariadb.org/browse/MDEV-9843>
 - Persistent statistics - <https://lists.launchpad.net/maria-discuss/msg04937.html>
- BLACKHOLE, FEDERATED (now FederatedX) require you to actually load the plugins
 - was a “surprise” - <https://jira.mariadb.org/browse/MDEV-11942> (now fixed)
- Status of other engines: OQGRAPH, SphinxSE? Cassandra deprecated

MyRocks

- RocksDB (Facebook) is a fork of LevelDB (Google). MyRocks is the interface to it from MySQL/MariaDB
- Write optimised
- Focus on the endurance of flash devices to gain better lifetime (10x less write amplification)
- Better compression than InnoDB (at least 2x)
- Ability to load data fast, avoiding compaction overheads
- Read-free replication (no random reads for updating secondary keys, only for unique indexes; RFR does away with it all, with row-based binlog)
- Recommended read: <https://mariadb.com/kb/en/library/differences-between-myrocks-variants/>

SPIDER

- Transparent sharding and re-sharding via SQL
- Partition by range/key/hash/list
 - vertical partitioning engine, allows partition by columns
- Condition pushdown to the storage engine layer
 - JOIN, GROUP BY done internally (on the data nodes/shards)
 - direct updates/deletes (pushdown to data nodes)
 - direct aggregates (sums, min, max, avg through partition engine)
- Partition improvements: full-text support, multi-range read (MRR)
- Read the docs, please! <https://mariadb.com/kb/en/library/spider-storage-engine-overview/>

Compression

- Row compression (ROW_FORMAT=COMPRESSED), to page compression (PAGE_COMPRESSED=1), now to column compression
- Bonus? Storage engine independent

```
CREATE TABLE `users` (  
  `id` int(11) NOT NULL,  
  `name` varchar(100) DEFAULT NULL,  
  `blurb` text /*!100301 COMPRESSED*/ DEFAULT NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=ROCKSDB DEFAULT CHARSET=latin1
```

Compression

```
show status like 'column_%';
```

Variable_name	Value
Column_compressions	1
Column_decompressions	0

Invisible columns

```
CREATE TABLE `user` (  
  `id` int(11) NOT NULL,  
  `name` varchar(100) DEFAULT NULL,  
  `secret` varchar(10) INVISIBLE DEFAULT  
  NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

```
insert into user (id,name,secret) values  
("1","colin","yes");
```

```
select * from user;
```

```
+-----+-----+  
| id | name |  
+-----+-----+  
|  1 | colin |  
+---+-----+
```

```
select id,name,secret from user;
```

```
+-----+-----+-----+  
| id | name | secret |  
+-----+-----+-----+  
|  1 | colin | yes    |  
+-----+-----+-----+
```

Open ended...

- How often are the 5.6/5.7 mysqltest's run on MariaDB Server?
- How often are upgrades from 5.6/5.7 tested?

System versioned tables

- SQL 2011 standard. Stores history of all changes.
- Can alter a table to enable/disable/remove system versioned data
- Queries?
 - AS OF to select data *as of* a point in time
 - BETWEEN .. AND to select data between *two points* in time
- Partition data BY SYSTEM_TIME
- Just ALTER .. ADD SYSTEM VERSIONING or create a table WITH SYSTEM VERSIONING

System versioned tables

```
create table employees (name varchar(10), salary int,  
department varchar(10)) with system versioning;
```

```
insert into employees values ("colin", 1000, "mktg");
```

```
update employees set salary=10000 where name="colin";
```

```
update employees set department="eng" where  
name="colin"
```

```
select * from employees where name="colin";
```

```
+-----+-----+-----+  
| name  | salary | dept |  
+-----+-----+-----+  
| colin | 10000 | eng  |  
+-----+-----+-----+
```

```
select *, ROW_START, ROW_END from employees for  
SYSTEM_TIME ALL;
```

```
+-----+-----+-----+  
+-----+-----+-----+  
+-----+  
| name  | salary | department | ROW_START  
| ROW_END |  
+-----+-----+-----+  
+-----+-----+-----+  
+-----+  
| colin | 10000 | eng        | 2018-06-26  
13:00:53.772241 | 2038-01-19 03:14:07.999999 |  
| colin | 1000  | mktg       | 2018-06-26  
13:00:03.656662 | 2018-06-26 13:00:24.251594 |  
| colin | 10000 | mktg       | 2018-06-26  
13:00:24.251594 | 2018-06-26 13:00:53.772241 |  
+-----+-----+-----+  
+-----+-----+-----+  
+-----+-----+-----+
```

AS OF example

```
SELECT * FROM employees FOR SYSTEM_TIME AS OF  
TIMESTAMP '2018-06-26 13:00:24';
```

```
+-----+-----+-----+  
| name   | salary | department |  
+-----+-----+-----+  
| colin  | 1000   | mktg       |  
+-----+-----+-----+
```

Oracle compatibility - Sequences

- Sequences to create a sequence of numeric values
- Not to be confused with replacing AUTO_INCREMENT, is an alternative to creating unique identifiers
- With a sequence, you can compute the last number created by all existing sequences, whereas AUTO_INCREMENT can only compute its own last number created

Sequences

```
create sequence seq start  
with 10 increment by 10;
```

```
select nextval(seq);
```

```
+-----+  
| nextval(seq) |
```

```
+-----+  
|      10 |
```

```
+-----+  
select nextval(seq);
```

```
+-----+  
| nextval(seq) |
```

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

```
|      20 |  
+-----+
```

```
select nextval(seq);
```

```
+-----+  
| nextval(seq) |
```

```
+-----+  
|      30 |
```

```
+-----+  
select lastval(seq);
```

```
+-----+  
| lastval(seq) |
```

```
+-----+  
|      30 |
```

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

```
select previous value for  
seq;
```

```
+-----+  
| previous value for seq |
```

```
+-----+  
|              30 |
```

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

Oracle PL/SQL

- PL/SQL compatibility parser added for easier migration from Oracle to MariaDB
- `sql_mode='oracle'`
- Data types (have synonyms in MariaDB): VARCHAR2 (VARCHAR), NUMBER (DECIMAL), DATE (DATETIME), RAW (VARBINARY), BLOB (LONGBLOB), CLOB (LONGTEXT)
- CURRVAL, NEXTVAL
- EXECUTE IMMEDIATE
- Existing stored procedures, triggers
- ROW datatype for stored routines
- Cursors with parameters
- Packages
- https://mariadb.com/kb/en/library/sql_modeoracle-from-mariadb-103/

Other bits

- INTERSECT and EXCEPT to UNION
- Stored aggregate functions - functions that are computed over a sequence of rows and return one result for the sequence of rows
 - see <https://jira.mariadb.org/browse/MDEV-16315>
- Idle transaction timeouts
 - `idle_transaction_timeout`, `idle_readonly_transaction_timeout`, `idle_write_transaction_timeout`

What about the rest?

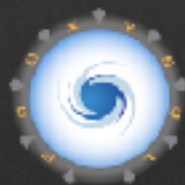
- Instant ADD COLUMN is in MySQL 8
- PROXY protocol support has been in Percona Server for MySQL

Some gotchas if you're in MySQL 8 land

- JSON is not stored as a binary data type
- GTIDs are different in MariaDB (e.g. no GTID in OK packet)
- No X Protocol, mysqlsh support
- No group replication
- PERFORMANCE_SCHEMA from MySQL 5.6
- No caching_sha256_password (ed25519)
- mysql.user.password now is mysql.user.authentication_string
- No password expiry, last changed, etc. however there is cracklib_password_check
- No optimiser hints, optimiser trace
- No SET PERSIST
- No Native data dictionary in MySQL 8 (atomic, crash-safe DDLs, faster INFORMATION_SCHEMA, no more MySQL system tables)
- Not as fast utf8mb4
- No persistent auto increment
- No automatically managed UNDO tablespace
- No InnoDB self-tuning (since InnoDB is from 5.7)
- No invisible indexes
- No TmpTable Storage engine
- No backup locks
- No InnoDB native partitioning
- No resource groups

An important enhancement in MySQL 5.7

- In MySQL 5.7 & Percona Server 5.7 an important feature was added which allows sending the GTID for a transaction on the OK packet for a transaction
- Enabled explicitly by setting `--session-track-gtids` to one of the following values:
 - "OWN_GTID": collect GTIDs generated for committed R/W transactions
 - "ALL_GTIDS": collect ALL GTIDs in `gtid_executed` when a R/W or R/O transaction commits
- Note: This feature is NOT available in MariaDB




Testing and QA matters

- <https://mariadb.org/docs/optimizing-tables/optimizing-tables-10-1-14/>
- <https://mariadb.org/docs/optimizing-tables/optimizing-tables-10-1-14/>

✓  **Kenny** added a comment - 2015-11-25 10:19

If MariaDB intended to implement this and set metadatalocks, I wonder what the real use case is here and maybe that should be updated in the documentation

- It is doing an 'offline' OPTIMIZE TABLE in a very very very slow manner, which does not interfere with workload to other tables, but the practical use is none IMO.
- The only real benefit here is that you can reduce/remove fragmentation on a table without having to create a new tablespace and need a lot of free disk space, but as it's so slow, it's not practical at all.

✓  **Sergei Golubchik** added a comment - 2016-05-02 20:35

This is a complex issue. OPTIMIZE always used to take rather strong locks because of MyISAM —MyISAM cannot allow any other connection accessing the table that is being optimized. InnoDB does not have this limitation. But we need to open the table to know whether it's MyISAM or InnoDB table. And we need to take metadata locks before the table is opened (because the table's engine is also part of metadata, we cannot read it without the metadata lock).

We'll fix it nevertheless. But may be this fix won't make it into 10.1.14

Today we already see this...

```
| Configuring mariadb-server-5.5 |
|
| MariaDB is a drop-in replacement for MySQL. It will use your current
| configuration file (my.cnf) and current databases.
|
| Note that MariaDB has some enhanced features, which do not exist in MySQL
| and thus migration back to MySQL might not always work, at least not as
| automatically as migrating from MySQL to MariaDB.
|
| Really migrate to MariaDB?
|
| <Yes> <No>
```

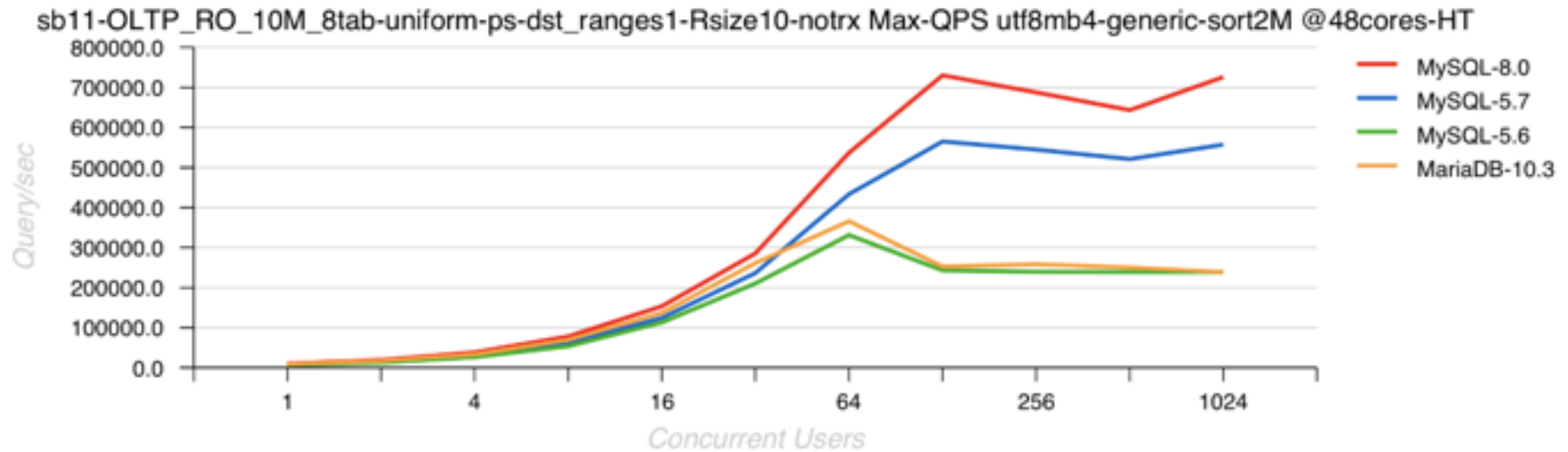
When to use MariaDB Server?

- More storage engines
 - MyRocks, TokuDB, CONNECT, SPIDER. MyISAM user? Segmented key caches will help
- Threadpool
- PAM authentication
- GSSAPI authentication (Kerberos, Active Directory)
- Optimistic parallel replication
- ANALYZE <statement>
- cracklib_password_check
- Oracle compatibility
- Temporal data (system versioned tables)
- PAM/GSSAPI/SSPI authentication
- AWS Key Management Plugin
- Table elimination
- User statistics
- Dynamic columns
- Invisible columns
- Query cache

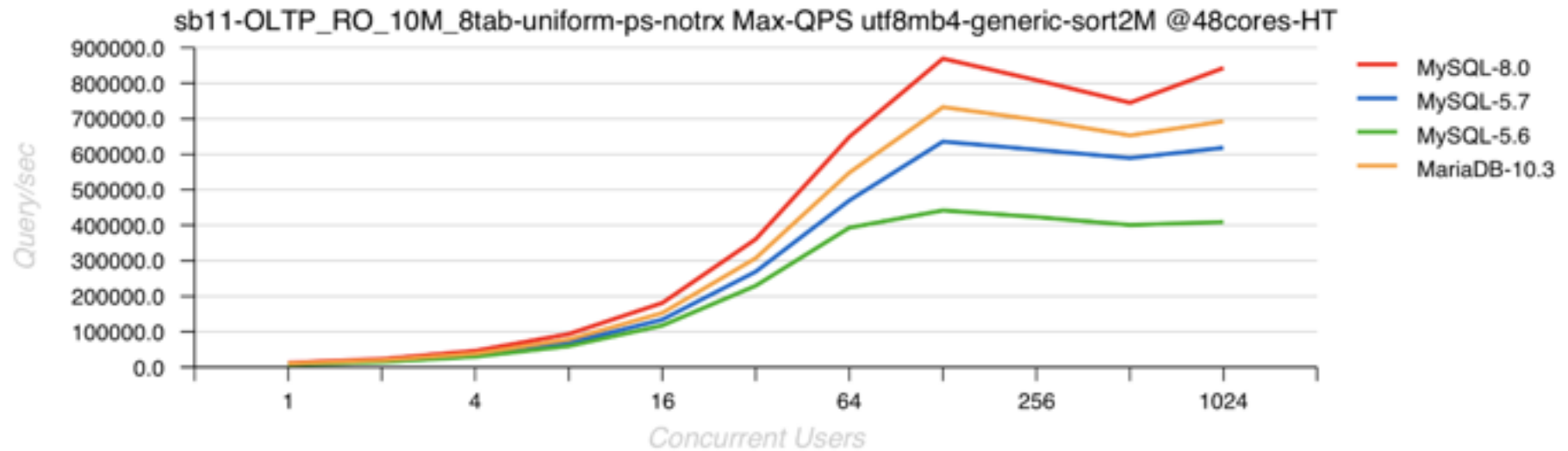
For most of everything else...

- <http://www.thecompletelistoffeatures.com/> (which is MySQL 5.7 based, but really, many of those features may not quite be in MariaDB Server 10.3 even)
- Performance...: <http://dimitrik.free.fr/blog/index.html>
- MySQL is making leaps and bounds around stability, High Availability, performance, security, observability & manageability
- think about a MySQL InnoDB Cluster, based on group replication that is configured in the MySQL Shell, with the MySQL Router? Try doing all this in MariaDB Server!

OLTP_RO 10Mx8tab "distinct-ranges" range=10 utf8mb4-generic :



OLTP_RO 10Mx8tab "mixed" utf8mb4-generic :



Thank you!

Colin Charles

colin.charles@percona.com / byte@bytebot.net

<http://bytebot.net/blog> | [@bytebot](#) on twitter

slides: slideshare.net/bytebot