MySQL Next –

An overview of the MySQL 8 Beta
Svan Lankes, Cocus AG
Carsten Thalheimer, Oracle MySQL
Agenda

• Timelines so far
• Features
• Performance
• MySQL vs. MariaDB vs. Percona

• Where to get
MySQL 8.0 RC1 Now Available!

MySQL 3.x ... MySQL 5.1
→ MySQL 5.5
→ MySQL 5.6
→ MySQL 5.7
→ MySQL 8.0

Wasn’t there a ‘6’ and ‘7’ between ???

Other MySQL Components will follow…

Major Release
- Every 2-3 years

Minor Releases
- Every Quarter
- Oracle CPU, Critical Patch Updates
- Security Alerts and Bulletins,
- e.g. 16. January 2018

Rapid Plug in deployments
- MySQL 8 has started uncoupling new features
- New features will be plug-ins (often) → new features quickly
MySQL 8.0 RC1 Now Available

What MySQL has done:
- 2 years in Development
- 400+ Worklogs
- 5000+ Bugs Fixed
- 500 new tests

Focus of MySQL 8:
- Mobile First
- Developer First
- Data Driven
- 24x7 at Scale
MySQL Innovation: 5.7 -> 8.0

2015
MySQL 5.7
- 3x Better Performance
- Replication Enhancements
- Optimizer Cost Model
- JSON Support
- Improved Security
- Sys & Performance Schema
- GIS

2016
MySQL 5.7 GR Plugin
- MySQL Group Replication
- MySQL X-Protocol (DMR)

2017
MySQL InnoDB Cluster
- MySQL Router
- MySQL Shell

2018
MySQL 8.0 (DMR)
- Data Dictionary
- Roles
- Unicode
- CTEs
- Window Functions
- Security
- Replication
- SysSchema
- GIS
MySQL Evolution


MySQL 8.0 (Labs, October 2014)
- MySQL Data Dictionary

MySQL 8.0.0 (DMR, September 2016) <Link>
- MySQL User roles
- MySQL 8.0 defaults to utf8mb4 & support Unicode 9.0
- SET PERSIST for global variables

MySQL 8.0.1 (DMR, April 2017) <Link>
- New JSON functions and improve performance and X-Protocol
- Common Table Expression
- Group Replication

MySQL 8.0.2 (DMR, Juli 2017) <Link>
- InnoDB Cluster capabilities
- SQL Window Functions
- New JSON functions

MySQL 8.0.3 (RC/Feature freeze, September 2017) <Link>
- Ressource Groups
- Autoscale InnoDB resources
MySQL Data Dictionary

- MySQL 5.7:
  - Schema/Databases → OS Directory, Table → 2x Files frm & ibd
    - frm, .par, .trg … Metadata per tables
    - ibd Data

- MySQL 8:
  - Schema/Databases → OS Directory, Table → 1x Files ibd (+ 1x sdi per schema)
    - sdi/InnoDB Metadata per schema
    - ibd Data
MySQL Data Dictionary

MySQL 5.7 (default) -> /var/lib/mysql

MySQL 8.0 (default) -> /var/lib/mysql
MySQL Data Dictionary

5.7:

Information_schema: Basically views, to read meta-data from Data Dictionary (> 50x faster)

8.0:
MySQL Data Dictionary – Why?

- More robust and more performant/scalable
- InnoDB (crash safe) holds metadata
- File corruption of a .frm file is not going to stall
- DDLs executed as internal atomic InnoDB Transaction
- Removed the file system's maximum number of files as the limiting factor
- you can now have literally have millions of tables in your database.
MySQL Evolution


MySQL 8.0 (Labs, October 2014)
- MySQL Data Dictionary

MySQL 8.0.0 (DMR, September 2016) <Link>
- MySQL User roles
- MySQL 8.0 defaults to utf8mb4 & support Unicode 9.0 Support
- SET PERSIST for global variables

MySQL 8.0.1 (DMR, April 2017) <Link>
- New JSON functions and improve performance and X-Protocol
- Common Table Expression
- Group Replication

MySQL 8.0.2 (DMR, Juli 2017) <Link>
- InnoDB Cluster capabilities
- SQL Window Functions
- New JSON functions

MySQL 8.0.3 (RC/Feature freeze, September 2017) <Link>
- Ressource Groups
- Autoscale InnoDB resources
Improving MySQL Access Controls

Easier to manage user and applications rights

As standards compliant as practically possible

Multiple default roles

Can export the role graph in GraphML

Example:

CREATE ROLE 'myrole';

GRANT DELETE, INSERT, SELECT, UPDATE, CREATE TEMPORARY TABLES ON *.* TO 'myrole';

CREATE USER 'donald'@'%' IDENTIFIED BY 'donald-passwd';

CREATE USER 'dagobert'@'localhost' IDENTIFIED BY 'dagobert-passwd';

GRANT 'myrole' TO 'donald'@'localhost'; GRANT 'myrole' TO 'dagobert'@'localhost';

SHOW GRANTS FOR 'donald'@'localhost';
UTF-8 / Unicode 9

The character set for the Web

MySQL 5.7: latin1 MySQL 8.0: utf8mb4

UTF-8 is the dominating character set in today’s applications

Requires 1-4 bytes for storing characters

Historically a performance problem

Example: emojis

![Emoji images](https://en.wikipedia.org/wiki/UTF-8)
MySQL 8.0 vs MySQL 5.7 utf8mb4

+300-350% in OLTP RO
+176-233% in OLTP RW
+1500-1800% in SELECT DISTINCT_RANGES
Persist Configuration

- Persist GLOBAL Server Variables
  - `SET PERSIST max_connections = 500;`
- Requires no filesystem access
- Includes timestamp and change user

- Examples Include:
  - Offline_mode
  - Read_Only
SET PERSIST for global variables

SET PERSIST innodb_buffer_pool_size = 250000000;
show variables like "innodb_buffer_pool_size";
→ 'innodb_buffer_pool_size', '268435456'

systemctl restart mysqld

more /var/lib/mysql/mysqld-auto.cnf
Find the source of variables changed on your installation

SELECT * FROM performance_schema.variables_info WHERE variable_source !~ 'COMPiled';

<table>
<thead>
<tr>
<th>VARIABLE_NAME</th>
<th>VARIABLE_SOURCE</th>
<th>VARIABLE_PATH</th>
<th>MIN_VALUE</th>
<th>MAX_VALUE</th>
<th>SET_TIME</th>
<th>SET_USER</th>
<th>SET_HOST</th>
</tr>
</thead>
<tbody>
<tr>
<td>autocommit</td>
<td>DYNAMIC</td>
<td>/etc/my.cnf</td>
<td>0</td>
<td>0</td>
<td>2017-11-20 15:12:44</td>
<td>root</td>
<td>localhost</td>
</tr>
<tr>
<td>datadir</td>
<td>GLOBAL</td>
<td></td>
<td>0</td>
<td>0</td>
<td>2017-11-20 15:20:15</td>
<td>root</td>
<td>localhost</td>
</tr>
<tr>
<td>foreign key checks</td>
<td>DYNAMIC</td>
<td></td>
<td>0</td>
<td>0</td>
<td>2017-11-20 15:11:44</td>
<td>root</td>
<td>localhost</td>
</tr>
<tr>
<td>innodb buffer pool size</td>
<td>PERSISTED</td>
<td>/var/lib/mysql/mysqld-auto.cnf</td>
<td>5242880</td>
<td>5223372036854775807</td>
<td>2017-11-20 15:11:44</td>
<td>root</td>
<td>localhost</td>
</tr>
<tr>
<td>log error</td>
<td>GLOBAL</td>
<td>/etc/my.cnf</td>
<td>0</td>
<td>0</td>
<td>2017-11-20 15:20:15</td>
<td>root</td>
<td>localhost</td>
</tr>
<tr>
<td>pid file</td>
<td>COMMAND LINE</td>
<td></td>
<td>0</td>
<td>0</td>
<td>2017-11-20 15:20:15</td>
<td>root</td>
<td>localhost</td>
</tr>
<tr>
<td>socket</td>
<td>GLOBAL</td>
<td>/etc/my.cnf</td>
<td>0</td>
<td>0</td>
<td>2017-11-20 15:20:15</td>
<td>root</td>
<td>localhost</td>
</tr>
<tr>
<td>sql_safe_updates</td>
<td>DYNAMIC</td>
<td>/etc/my.cnf</td>
<td>0</td>
<td>0</td>
<td>2017-11-20 15:12:44</td>
<td>root</td>
<td>localhost</td>
</tr>
<tr>
<td>transaction isolation</td>
<td>DYNAMIC</td>
<td>/etc/my.cnf</td>
<td>0</td>
<td>0</td>
<td>2017-11-20 15:12:44</td>
<td>root</td>
<td>localhost</td>
</tr>
</tbody>
</table>
MySQL Evolution

MySQL 8.0 (Labs, October 2014)
- MySQL Data Dictionary

MySQL 8.0.0 (DMR, September 2016) <Link>
- MySQL User roles
- MySQL 8.0 defaults to utf8mb4 & support Unicode 9.0
- SET PERSIST for global variables

MySQL 8.0.1 (DMR, April 2017) <Link>
- New JSON functions and improve performance and X-Protocol
- Common Table Expression
- Group Replication

MySQL 8.0.2 (DMR, Juli 2017) <Link>
- InnoDB Cluster capabilities
- SQL Window Functions
- New JSON functions

MySQL 8.0.3 (RC/Feature freeze, September 2017) <Link>
- Ressource Groups
- Autoscale InnoDB resources
Javascript Everywhere

Backend:

node + express

Frontend:

AngularJS + Vue + React
<table>
<thead>
<tr>
<th>JSON Function</th>
<th>JSON Function</th>
<th>JSON Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSON_ARRAY_APPEND()</td>
<td>JSON_MERGE_FREE()</td>
<td>JSON_VALID()</td>
</tr>
<tr>
<td>JSON_ARRAY_INSERT()</td>
<td>JSON_MERGE_FREE()</td>
<td>JSON_PRETTY()</td>
</tr>
<tr>
<td>JSON_ARRAY()</td>
<td>JSON_OBJECT()</td>
<td>JSON_STORAGE_SIZE()</td>
</tr>
<tr>
<td>JSON_CONTAINS_PATH()</td>
<td>JSON_QUOTE()</td>
<td>JSON_STORAGE_FREE()</td>
</tr>
<tr>
<td>JSON_CONTAINS()</td>
<td>JSON_REMOVE()</td>
<td>JSON_ARRAYAGG()</td>
</tr>
<tr>
<td>JSON_DEPTH()</td>
<td>JSON_REPLACE()</td>
<td>JSON_OBJECTAGG()</td>
</tr>
<tr>
<td>JSON_EXTRACT()</td>
<td>JSON_SEARCH()</td>
<td>JSON_MERGE_PATCH()</td>
</tr>
<tr>
<td>JSON_INSERT()</td>
<td>JSON_SET()</td>
<td>JSON_TABLE() *labs</td>
</tr>
<tr>
<td>JSON_KEYS()</td>
<td>JSON_TYPE()</td>
<td></td>
</tr>
<tr>
<td>JSON_LENGTH()</td>
<td>JSON_UNQUOTE()</td>
<td></td>
</tr>
</tbody>
</table>
Common Table Expressions

“With queries”
- CTEs are statement duration views.
- Inner view here does not have to materialize it,
- Merge it just like a view
- Both Recursive and Non-Recursive Forms (A recursive CTE refers to itself in a subquery)
- Similar to Oracle's CONNECT BY

**Simplifies writing complex SQL:**

```sql
WITH myt1 AS (SELECT * FROM tblA WHERE a='b') SELECT * FROM myt1;
```

*e.g.*

```sql
WITH german_sales AS (SELECT * FROM sales WHERE country='DE'),
monthly_sales AS (SELECT MONTH(created_at) mon, SUM(price) total_price FROM german_sales group by 1)
SELECT MONTH(created_at), price/total_price FROM german_sales JOIN monthly_sales ON mon=month(created_at);
```
MySQL Group Replication: Database HA

Group Replication library

- Implementation of Replicated Database State Machine
  - MySQL GCS is based on our home-grown Paxos implementation
- Provides virtually synchronous replication for MySQL 5.7+
  - Guarantees eventual consistency
- Automates operations
  - Conflict detection and resolution
  - Failure detection, fail-over, recovery
  - Group membership management and reconfiguration

“Multi-master update anywhere replication plugin for MySQL with built-in conflict detection and resolution, automatic distributed recovery, and group membership.”
MySQL Shell: DBA Admin API

The global variable 'dba' is used to access the MySQL AdminAPI

mysql-js> dba.help()

Perform DBA operations

• Manage MySQL InnoDB clusters
  • Create clusters
  • Validate MySQL instances
  • Configure MySQL instances
  • Get cluster info
  • Modify clusters
  • and much more ...
What’s New in the MySQL Shell

Customizable prompt
  • Include context and session information

Custom font and color support

Persistent command line history

Auto-complete / Content Assistance

Full Unicode support
“High Availability becomes a core first class feature of MySQL!”
MySQL Evolution


MySQL 8.0 (Labs, October 2014)
- MySQL Data Dictionary

MySQL 8.0.0 (DMR, September 2016) <Link>
- MySQL User roles
- MySQL 8.0 defaults to utf8mb4 & support Unicode 9.0
- SET PERSIST for global variables

MySQL 8.0.1 (DMR, April 2017) <Link>
- New JSON functions and improve performance and X-Protocol
- Common Table Expression
- Group Replication

MySQL 8.0.2 (DMR, Juli 2017) <Link>
- InnoDB Cluster capabilities
- SQL Window Functions
- New JSON functions

MySQL 8.0.3 (RC/Feature freeze, September 2017) <Link>
- Ressource Groups
- Autoscale InnoDB resources
MySQL 8.0 RC: Resource Group Example

System Configuration:
Oracle Linux 7,
Intel(R) Xeon(R) CPU E7-4860 2.27GHz
40 cores-HT

<table>
<thead>
<tr>
<th>Queries per Second</th>
<th>No Resource Group (40 Cores Shared)</th>
<th>With Resource Group (40 Cores for Select) (10 Cores for Update RG)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td><img src="#" alt="Select Bar" /></td>
<td><img src="#" alt="Select Bar" /></td>
</tr>
<tr>
<td>Update</td>
<td><img src="#" alt="Update Bar" /></td>
<td><img src="#" alt="Update Bar" /></td>
</tr>
</tbody>
</table>

Select and Update queries per second comparison with and without resource group.
MySQL Evolution


MySQL 8.0 (Labs, October 2014)
- MySQL Data Dictionary

MySQL 8.0.0 (DMR, September 2016) <Link>
- MySQL User roles
- MySQL 8.0 defaults to utf8mb4 & support Unicode 9.0
- SET PERSIST for global variables

MySQL 8.0.1 (DMR, April 2017) <Link>
- New JSON functions and improve performance and X-Protocol
- Common Table Expression
- Group Replication

MySQL 8.0.2 (DMR, Juli 2017) <Link>
- InnoDB Cluster capabilities
- SQL Window Functions
- New JSON functions

MySQL 8.0.3 (RC/Feature freeze, September 2017) <Link>
- Ressource Groups
- Autoscale InnoDB resources

And much more ...
MySQL 8.0: Makes Managing Scale Easier

Greater density per server
Operator Friendliness
Increased Reliability
New! Improved Replication Throughput
Performance Improvements

Improved Query Consistency
  • Histograms
  • Improved Cost Model

Faster Table/Range Scans

• Parallel Replication
• UTF8MB4
• Information Schema
• Performance Schema Indexes
MySQL 8.0: Parallel Replication

Production Workload #1

Production Workload #2

4x Improvement on single-threaded
InnoDB Auto Increment Persists

- First reported as BUG #199
- Auto increment counters are now written to the REDO log
- Allows for fast changing meta data
Descending Indexes

For B+tree indexes

CREATE TABLE t1 (  
a INT,  
b INT,  
INDEX a_b (a DESC, b ASC));

In 5.7: Index in ascending order is created, server scans it backwards
In 8.0: Index in descending order is created, server scans it forwards

Benefits:
Forward index scan is faster than backward index scan
Use indexes instead of filesort for ORDER BY clause with ASC/DESC sort key
New! InnoDB Redo and Undo Encryption

AES 256 encryption

*Encrypted when redo/undo log data is written to disk*

Decryption occurs when redo/undo log data is read from disk

Once redo/undo log data is read into memory, it is in unencrypted form.

Two tiered encryption – like Innodb tablespace encryption

  • Fast key rotation, high performance

*Easy to use*

  • Enabled using `innodb_redo_log_encrypt` and `innodb_undo_log_encrypt`
New! Invisible Indexes

Indexes are “hidden” to the MySQL Optimizer
  • Not the same as “disabled indexes”
  • Contents are fully up to date and maintained by DML

Two use cases:
  • Soft Delete (Recycle Bin)
  • Staged Rollout

→ I don’t think this index is used any more:
  ALTER TABLE Country ALTER INDEX c INVISIBLE;

→ I need to revert:
  ALTER TABLE Country ALTER INDEX c VISIBLE;

→ It is now safe to drop:
  ALTER TABLE Country DROP INDEX c;
Evolution from MySQL 5.5 to 8.0

MySQL 5.5
- Event Waits
- Mutexes
- Files
- Threads

MySQL 5.6
- Statement Instrumentation
- Lower Overhead

MySQL 5.7
- Memory Instrumentation
- Prepared Statements Instrumentation
- Transactions Instrumentation
- Scalable Memory Allocation
- Bundled SYS schema
- Lower Overhead

MySQL 8.0
- Histograms
- Indexes
- Data Locks instrumentation
- SQL Errors instrumentation
- Variables Instrumentation
- Table plugin
- Improved Defaults
Performance Schema Indexes

Allows for more efficient access to Performance Schema tables

A total of **90 indexes** across **89 tables**

Adds zero overhead

• A physical index is not maintained internally
• Implementation of indexes *tricks* the optimizer into better execution plan
Information Schema Performance

100 schemas times 50 tables (5000 tables)

- Count All Schemas
- Schema aggregate size stats
- All Dynamic Table Info
- All Static Table Info
- Auto Increments Near Limit
- Count All Columns
- Count All Indexes

Time in Seconds (Lower is better)
New! Better Handling of Hot Row Contention

One of the id 1, 2, 3, 4 is locked... what if?

- SELECT * FROM tickets WHERE id IN (1,2,3,4) AND order_id IS NULL FOR UPDATE;
- Error immediately if a row is already locked
- Non deterministically skip over locked rows
- SELECT * FROM tickets WHERE id IN (1,2,3,4) AND order_id IS NULL FOR UPDATE NOWAIT;
- Timeout after 50 sec (default)
- SELECT * FROM tickets WHERE id IN (1,2,3,4) AND order_id IS NULL FOR UPDATE SKIP LOCKED;
• Binary format is now smaller and insert-order efficient:

From `VARCHAR(36)` 53303f87-78fe-11e6-a477-8c89a52c4f3b

To `VARBINARY(16)` 11e678fe53303f87a4778c89a52c4f3b
Evolution continued in MySQL 8.0

<table>
<thead>
<tr>
<th>Geography support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spatial Reference Systems (SRS) Support</td>
</tr>
<tr>
<td>SQL/MM Information Schema views</td>
</tr>
<tr>
<td>Standard compliant axis ordering in import/export functions</td>
</tr>
<tr>
<td><strong>Utf8mb4 as default</strong></td>
</tr>
<tr>
<td>Up to 16x Faster Performance</td>
</tr>
<tr>
<td>Based on Unicode 9.0</td>
</tr>
<tr>
<td>New collations with Accent/Case sensitivity</td>
</tr>
<tr>
<td>Japanese Collation</td>
</tr>
<tr>
<td><strong>Aggregate, table and Pretty print Functions</strong></td>
</tr>
<tr>
<td>Sort Performance Improvements</td>
</tr>
<tr>
<td>Optimizer support for in-place update</td>
</tr>
<tr>
<td>JSON Shell</td>
</tr>
<tr>
<td><strong>Roles</strong></td>
</tr>
<tr>
<td>Encrypted REDO Logs and UNDO space</td>
</tr>
<tr>
<td>Finer grained system privileges</td>
</tr>
<tr>
<td>Atomicity in Privilege statements</td>
</tr>
</tbody>
</table>
All these features plus…

Source code now documented with Doxygen
Plugin Infrastructure!
Expanded GIS Support
Expanded Query Hints Support
Improved Scan Query Performance
Improved BLOB Storage
Improved Memcached Interface
Cost Model Improvements
Scalability Improvements
Atomicity in Privileges
Parser Refactoring
Improvements to Temporary Tables
C++11 and Toolchain Improvements

- GTID_PURGED always settable
- Persistent Auto Increment
- Native InnoDB Data dictionary
- Improved Information Schema performance
- New! SQL Grouping Function
- New! Optimizer Trace detailed sort statistics
- New! Descending Indexes
- New! Smaller Package Downloads
- New! JSON Aggregate, Pretty print functions
- New! JSON performance improvements
- New! Expanded Query Hints
- New! Improved usability of cost constant configuration

- Group Replication in 8.0
- New! Transaction Save Point support in Group Replication
- New! Improved Replication Applier Lag Statistics in Performance Schema
- New! Per Multi-source Channel Replication Filters
- New! Atomic DDL extended to the Binary Log
- New! Performance Improvements on the Replication Applier
- New! Parallel Replication Applier Policy
- New! Binary Log Management Enhancements
- New! Additional Metadata Into the Binary Log
Thank you!

Merci!

Vielen Dank!

Sven Lakes
slankes@cocus.com
M:

Carsten Thalheimer
Carsten.Thalheimer@Oracle.com
M: +49 172 / 8833386