MySQL 8.0
New Features

DOAG K&A 2018, Nürnberg

Oli Sennhauser
Senior MySQL Consultant at FromDual GmbH

https://www.fromdual.com/presentations
MySQL 8.0 – New Features

➢ History
➢ Transactional Data Dictionary
➢ Security and Account Management
➢ Resource Management
➢ InnoDB Enhancements
➢ Window Functions
➢ Common Table Expressions (CTE)
➢ Other New Features
➢ Upgrade to MySQL 8.0
History

- 5.0 – 2005 – SQL/PSM, I_S, XA
- 5.1 – 2008 – Partitions, Row Based Replication
- 5.5 – 2010 – InnoDB default, Scalability
- 5.6 – 2013 – P_S, Security, GTID, Online DDL
- 5.7 – 2015 – JSON, Spatial, Group Replication, Security
- 8.0 – 2018 – ?

- Every 3rd year a new major release
Transactional Data Dictionary

- Transactional DD, stored INSIDE MySQL
- Before:
  - External Metadata Files (.frm, ...)
  - mysql.* MyISAM tables
  - InnoDB internal DD (I_S.innodb_sys_*)
- Now:
  - All InnoDB, mysql.ibd Tablespace :-)
  - Removed: .frm .par .TRN .TRG .isl db.opt :-)
  - DD tables protected (hidden) by default :-(
  - INFORMATION_SCHEMA tables renamed and columns capitalized! :-(
- Atomic DDL (single atomic DDL transaction)
  - DD Update + SE Operation + Binary Log write
  - InnoDB only
Transaction Data Dictionary

SQL> SHOW GLOBAL VARIABLES LIKE '%cache';
+---------------------------------+-------+
| Variable_name                   | Value |
+---------------------------------+-------+
| schema_definition_cache         | 256   |
| stored_program_definition_cache | 256   |
| table_definition_cache          | 1400  |
| tablespace_definition_cache     | 256   |
+---------------------------------+-------+

No status... :-(

SQL> SET SESSION debug='+d,skip_dd_table_access_check';
ERROR 1193 (HY000): Unknown system variable 'debug'

SQL> SELECT name, schema_id, hidden, type
 FROM mysql.tables WHERE schema_id=1 AND hidden='System';
ERROR 3554 (HY000): Access to data dictionary table
'mysql.tables' is rejected.

SQL> SHOW CREATE TABLE mysql.catalogs\G
ERROR 3554 (HY000): Access to data dictionary table
'mysql.catalogs' is rejected.
Operational impact of Trx DD

- *innodb_read_only*
  - Also affects non-InnoDB tables
  - Also affects `ANALYZE TABLE`
  - Also affects `ALTER TABLE ... ENGINE=...`
- System Statistics are cached (Optimizer)
  - Update → `ANALYZE TABLE`
  - `information_schema_stats_expiry = 86400`
- `CREATE TABLE ... LIKE` must be a base table (no VIEW or I_S table)
- I_S tables are partly renamed → Some applications might not work any more...
- I_S table columns are capitalized → Some applications might not work any more...
- Backup: `mysqldump --all-databases --routines --events MUST BE used now!!!`
  - What about --triggers???
  - FromDual recommends since long...!
- Manual creation of a Schema (`mkdir`) is NOT supported any more!
- DDL operations take longer (`mysqldump!!! How many tables do you have??`)
- Upgrade! See later...
Security and Account Mgmt

• New authentication plug-in:
  • caching_sha2_password (SHA-256 + caching)
  • Addresses latency issues at connect
  • More connection protocols
  • No linking against OpenSSL required
  • Default (rather than mysql_native_password)

• Needs client side plug-in (8.0)
  • libmysqlclient
  • Pre 8.0 client: mysql_native_password
  • Pre 5.7 must be upgraded
Client:

shell> mysql --user=app --password=secret
ERROR 2059 (HY000): Plugin caching_sha2_password could not be loaded: caching_sha2_password.so: cannot open shared object file: No such file or directory

Server:

[Note] [MY-010914] [Server]
Got an error reading communication packets

Application:

shell> ./insert_test.php
/usr/bin/php: libcrypto.so.1.0.0: no version information available (required by /usr/bin/php)
PHP Warning: mysqli::mysqli(): (HY000/2059): Authentication plugin 'caching_sha2_password' cannot be loaded: caching_sha2_password.so: cannot open shared object file: No such file or directory in test.php on line 13
ROLEs – Admin side

SQL> CREATE ROLE 'ro_user';
ERROR 1396 (HY000): Operation CREATE ROLE failed for 'ro_user'@'%'

SQL> CREATE ROLE 'ro_role';
SQL> GRANT SELECT ON test.* TO 'ro_role';

SQL> SHOW GRANTS FOR 'ro_role';
+-------------------------------------------+
| Grants for ro_role@%                      |
+-------------------------------------------+
| GRANT USAGE ON *.* TO `ro_role`@`%`      |
| GRANT SELECT ON `test`.* TO `ro_role`@`%` |
+-------------------------------------------+

SQL> GRANT 'ro_role' to 'ro_user'@'%';

SQL> -- SET DEFAULT ROLE 'ro_role' TO 'ro_user'@'%';

[mysqld]
mandatory_roles='ro_role'
ROLEs – User side

SQL> SELECT CURRENT_ROLE();
+----------------+
| CURRENT_ROLE() |
+----------------+
| NONE           |
+----------------+

SQL> SET ROLE 'ro_role';

SQL> SHOW GRANTS FOR 'ro_user'@'%;
+--------------------------------------+
| Grants for ro_user@%                |
+--------------------------------------+
| GRANT USAGE ON *.* TO `ro_user`@`%`  |
| GRANT `ro_role`@`%` TO `ro_user`@`%` |
+--------------------------------------+

SQL> SET ROLE ALL;

SQL> SHOW GRANTS FOR CURRENT_USER() USING 'ro_role';
Resource Management

- Define Resource Groups (RG)
  - Assigning Threads/Connections/Queries to Resource Groups
- CPU time is a manageable Resource "virtual CPU"
  - CPU only atm.
- Resource Group types:
  - `SYSTEM` (background threads), Prio: -20 to 0
  - `USER` (foreground/user threads), Prio: 0 to 19
- Default groups:
  - `SYS_default`, no CPU Affinity and Priority 0
  - `USR_default`, no CPU Affinity and Priority 0
- CPU Affinity: Set of virtual CPU a Resource Group can use
- Thread Priority: -20 (highest) to 19 (lowest), default 0
RG – Admin Side

- Privilege: RESOURCE_GROUP_ADMIN (new)

CREATE RESOURCE GROUP batch
  TYPE = USER
  VCPU = 2-3
  THREAD_PRIORITY = 10
;
Query OK, 0 rows affected, 1 warning (0.02 sec)

SQL> show warnings;
+--------------------------+-----------------+---------------------------------------+
| Level | Code | Message                               |
+--------------------------+-----------------+---------------------------------------+
| Warning | 3659 | Attribute thread_priority is ignored. |
+--------------------------+-----------------+---------------------------------------+
shell> sudo setcap cap_sys_nice+ep /opt/mysql-8.0/bin/mysqld

shell> getcap /opt/mysql-8.0/bin/mysqld
/opt/mysql-8.0/bin/mysqld = cap_sys_nice+ep
shell> systemctl restart mysqld

SystemD:

[Service]
AmbientCapabilities=CAP_SYS_NICE

shell> ps -L -eo pid,tid,ni,pcpu,stat \ 
    | grep -e 14078 -e PID | cut -b-160
    PID   TID   NI  %CPU STAT
14078 14078  0   0.0 S l
... 14078 15626  10  0.0 SN1 14078 14316  0   0.0 S l
RG Information

SQL> SELECT * FROM INFORMATION_SCHEMA.RESOURCE_GROUPS;
+-------------+---------+---------+----------+----------+
| NAME        | TYPE    | ENABLED | VCPU_IDS | PRIORITY |
+-------------+---------+---------+----------+----------+
| USR_default | USER    |       1 | 0-3      |        0 |
| SYS_default | SYSTEM  |       1 | 0-3      |        0 |
| batch       | USER    |       1 | 2-3      |       10 |
+-------------+---------+---------+----------+----------+

SQL> SELECT thread_id, name, type, thread_os_id, resource_group FROM performance_schema.threads;
+-----------+--------------------------+------------+-----------+----------------+
| thread_id | name                     | type       | os_thread | resource_group |
+-----------+--------------------------+------------+-----------+----------------+
|         1 | sql/main                 | BACKGROUND |     14078 | SYS_default    |
|         3 | innodb/io_ibuf_thread    | BACKGROUND |     14081 | SYS_default    |
|         4 | innodb/io_log_thread     | BACKGROUND |     14082 | SYS_default    |
| ...      |                          |            |          |                |
|         41| innodb/srv_worker_thread | BACKGROUND |     14119 | SYS_default    |
|         42| sql/event_scheduler      | FOREGROUND |     14123 | SYS_default    |
|         44| sql/compress_gtid_table | FOREGROUND |     14125 | SYS_default    |
|         566| sql/one_connection       | FOREGROUND |     15626 | batch          |
|         457| sql/one_connection       | FOREGROUND |     14126 | USR_default    |
+-----------+--------------------------+------------+-----------+----------------+
RG – User Side

- Privilege: RESOURCE_GROUP_USER (new)

```
SET RESOURCE GROUP batch FOR /* thread */ 42;
--

SQL> SET RESOURCE GROUP batch;
ERROR 1227 (42000): Access denied; you need (at least one of) the RESOURCE_GROUP_ADMIN OR RESOURCE_GROUP_USER privilege(s) for this operation

SQL> -- GRANT RESOURCE_GROUP_USER ON *.* TO 'ro_user'@'\%';

SQL> SELECT /*+ RESOURCE_GROUP(batch) */ * FROM test.test;
```
RG Restrictions / Use-cases

- Restrictions
  - (Enterprise) Thread Pool plug-in does NOT work (shared threads)
  - Mac OSX: Not available (kernel does not provide API)
  - FreeBSD and Solaris: Thread priorities are ignored
  - Linux: `CPU_SYS_NICE` capability must be set (SystemD)
  - Windows: Thread Priority mapping (Windows has only 5)

- Use-cases:
  - CPU over-provisioning...
  - Uncontrolled application behaviour...
  - Cloud?

- Not really easy (KISS) and thus potential dangerous!
InnoDB enhancements I

- A lot of small enhancements for specific use-cases:
  - AUTO_INCREMENT handling
  - Disable deadlock detection (`innodb_deadlock_detect`)
  - Temporary table handling
    - New Storage Engine
    - BLOB support
    - Pool of temporary tablespaces
  - InnoDB encryption for redo log, undo log and general TS
  - InnoDB supports NOWAIT and SKIP LOCKED with
    - `SELECT ... FOR SHARE`
    - `SELECT ... FOR UPDATE`
InnoDB enhancements II

- Native Partition handling
  - → no more MyISAM Partitions! :-(
- UNDO Tablespace management improved
  - Write throughput
- Renaming a General Tablespace is supported
  - `ALTER TABLESPACE ... RENAME TO`
- Moving Tablespace Files while the Server is offline
- Auto configuration: `innodb_dedicated_server`
  - `innodb_buffer_pool_size`, `innodb_log_file_size`, `innodb_flush_method`
- Redo logging optimizations
  - Write throughput
- Instantaneous `ALTER TABLE` support: `ALGORITHM=INSTANT`
  - `ADD COLUMN`, `ADD/DROP` Virtual column, `ADD/DROP` default value, change enum/set definition, `RENAME TABLE`
- Parallel index reads (begin of parallel query option?)
  - non-locking `SELECT COUNT(*)` and `CHECK TABLE`
Character Set

- New default: utf8mb4
- Now we can also store...
  - ...in the database!
- utf8(mb3) deprecated!
- What about query latency???
XPath expressions:

- \( \text{->>} \) (in-line path) operator → \( \text{JSON\_UNQUOTE(\text{JSON\_EXTRACT(json))}} \)
- Ranges such as \( \{1 \text{ to } 5\} \)
- \( \{\text{last-2 to last-1}\} \)

New functions: \( \text{JSON\_ARRAY\_AGG()} \), \( \text{JSON\_OBJECT\_AGG()} \), \( \text{JSON\_PRETTY()} \), \( \text{JSON\_STORAGE\_SIZE()} \), \( \text{JSON\_STORAGE\_FREE()} \), \( \text{JSON\_TABLE()} \)

\text{ORDER BY} on JSON documents more efficient

Partial, in-place update of JSON: \( \text{JSON\_SET()} \), \( \text{JSON\_REPLACE()} \), \( \text{JSON\_REMOVE()} \)

RFC 7396 implementation of \( \text{JSON\_MERGE\_PATCH()} \)

Implemented "last duplicate key wins" normalization of duplicate keys
- consistent with RFC 7159 and most JavaScript parsers
- old: "first duplicate key wins" (incompatibility change!!!)
Optimizer

• Invisible Indexes (not visible to the Optimizer)
  • `CREATE INDEX invi_ind
      ON test_tab (data_col) INVISIBLE;`

• Descending (B-Tree) Indexes for ORDER BY optimization
  • `... ADD INDEX dsc_i (ts DESC, name ASC)`
  • `... ORDER BY ts DESC, name ASC`

• Functional Index Key parts (Function Based Indexes)
  • `CREATE INDEX fbi1 ON test_tab ((UPPER(name)));`
  • `CREATE INDEX fbi2 ON t1 ((col1 + col2), (col1 - col2), col1);`
  • `ALTER TABLE test_tab ADD INDEX ((col1 * 40) DESC);`
Explicit DEFAULT Functions

- Support for DEFAULT for table columns with:
  - Constant or Expressions/Functions
- Referring to earlier columns in the DEFAULT expression is possible.
- The DEFAULT clause CANNOT contain any stored functions or subqueries.
CREATE TABLE employee (
  id INT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
  uuid VARCHAR(48) DEFAULT (UUID()),
  first_name VARCHAR(30) NOT NULL,
  last_name VARCHAR(40) NOT NULL,
  salary DECIMAL(7,2) DEFAULT 1000.0,
  bonus DECIMAL(7, 2) DEFAULT (salary * 0.1)
);

INSERT INTO employee
VALUES (DEFAULT, DEFAULT, 'Oli', 'Sennhauser', 900.0, DEFAULT);

SELECT * FROM employee;

+----+--------------------+------------+------------+--------+-------+
| id | uuid               | first_name | last_name  | salary | bonus |
+----+--------------------+------------+------------+--------+-------+
|  1 | 601964996657676299 | Oli        | Sennhauser | 900.00 | 90.00 |
+----+--------------------+------------+------------+--------+-------+

ALTER TABLE employee
MODIFY bonus DECIMAL(7, 2) DEFAULT (salary * 0.12);
Window Functions

• What is it?
  • Buzzwords: Reporting, DWH, OLAP, Cube, Business Intelligence, Analytics, Big Data
  • SQL:2003 – SQL:2011

• For each row from a query
  • Perform calculations using rows related to that row

  \texttt{window\_function(expression) \ OVER \ ( \ PARTITION \ BY \ ... \ ORDER \ BY \ ...)}

• Window Functions are computed after all \texttt{WHERE, GROUP \ BY} and \texttt{HAVING} clauses, right before \texttt{ORDER \ BY}
Available Window Functions

- Ranking, Bucketing
  - `DENSE_RANK`, `NTILE`, `NTH_VALUE`, `PERCENT_RANK`, `RANK`, `ROW_NUMBER`

- References other rows/data
  - `FIRST_VALUE`, `LAG`, `LAST_VALUE`, `LEAD`

- Statistics
  - `CUME_DIST`

- MySQL 8 also supports Window Frames
  - `{ RANGE | ROWS } BETWEEN frame_start AND frame_end`
  - `frame = {value PRECEDING | CURRENT ROW | value FOLLOWING}`
Window Functions Example

- Top 3 earners of each department and compare salaries with the average salary

```
SELECT *
FROM (  
    SELECT RANK() OVER (PARTITION BY dept ORDER BY salary DESC) AS ranking, dept, name, salary  
    FROM employee_salaries  
    , ROUND(AVG(salary) OVER (PARTITION BY dept), 2) AS avg_salary  
    ) AS salary_ranks
WHERE ranking <= 3;
```

<table>
<thead>
<tr>
<th>ranking</th>
<th>dept</th>
<th>name</th>
<th>salary</th>
<th>avg_salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Engineering</td>
<td>Dharma</td>
<td>3500</td>
<td>2633.33</td>
</tr>
<tr>
<td>2</td>
<td>Engineering</td>
<td>Bình</td>
<td>3000</td>
<td>2633.33</td>
</tr>
<tr>
<td>3</td>
<td>Engineering</td>
<td>Adalynn</td>
<td>2800</td>
<td>2633.33</td>
</tr>
<tr>
<td>1</td>
<td>Sales</td>
<td>Carbry</td>
<td>500</td>
<td>325.00</td>
</tr>
<tr>
<td>2</td>
<td>Sales</td>
<td>Clytemnestra</td>
<td>400</td>
<td>325.00</td>
</tr>
<tr>
<td>3</td>
<td>Sales</td>
<td>Juraj</td>
<td>300</td>
<td>325.00</td>
</tr>
<tr>
<td>3</td>
<td>Sales</td>
<td>Kalpana</td>
<td>300</td>
<td>325.00</td>
</tr>
</tbody>
</table>
```
Common Table Expressions

- **WITH** → Common Table Expression (CTE)
  - Refer to a sub-query expression many times
  - Like a temporary table per query
  - Temporary named result set / View
  - Makes SQL more readable
  - Since SQL:1999

- **Non-recursive CTE**
- **Recursive CTE**

```sql
WITH [ RECURSIVE ] cte AS
  ( SELECT * FROM test WHERE id = 42 )
SELECT * FROM cte;
```
WITH SalesCTE (SalesPersonID, SalesOrderID, SalesYear) AS 
(
    SELECT SalesPersonID, SalesOrderID, YEAR(OrderDate) AS SalesYear
    FROM SalesOrderHeader
    WHERE SalesPersonID IS NOT NULL
)
SELECT SalesPersonID, COUNT(SalesOrderID) AS TotalSales, SalesYear
FROM SalesCTE
GROUP BY SalesYear, SalesPersonID
ORDER BY SalesPersonID, SalesYear;

+---------------+------------+-----------+
<table>
<thead>
<tr>
<th>SalesPersonID</th>
<th>TotalSales</th>
<th>SalesYear</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2018</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>2018</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2018</td>
</tr>
</tbody>
</table>
+---------------+------------+-----------+
WITH RECURSIVE DirectReports(Name, Title, EmployeeID, EmployeeLevel, Sort) AS
(
    SELECT CONCAT(e.FirstName, ' ', e.LastName),
           e.Title, e.EmployeeID, 1,
           CONCAT(e.FirstName, ' ', e.LastName)
    FROM MyEmployees AS e
    WHERE e.ManagerID IS NULL
    UNION ALL
    SELECT CONCAT(REPEAT('|    ', EmployeeLevel), e.FirstName, ' ', e.LastName),
           e.Title, e.EmployeeID, EmployeeLevel + 1,
           CONCAT(RTRIM(Sort), '|    ', FirstName, ' ', LastName)
    FROM MyEmployees AS e
    JOIN DirectReports AS d ON e.ManagerID = d.EmployeeID
)
SELECT EmployeeID, Name, Title, EmployeeLevel
FROM DirectReports
ORDER BY Sort;

<table>
<thead>
<tr>
<th>EmployeeID</th>
<th>Name</th>
<th>Title</th>
<th>EmployeeLevel</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ken Sánchez</td>
<td>Chief Executive Officer</td>
<td>1</td>
</tr>
<tr>
<td>273</td>
<td>Brian Welcker</td>
<td>Vice President of Sales</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>David Bradley</td>
<td>Marketing Manager</td>
<td>3</td>
</tr>
<tr>
<td>23</td>
<td>Mary Gibson</td>
<td>Marketing Specialist</td>
<td>4</td>
</tr>
<tr>
<td>274</td>
<td>Stephen Jiang</td>
<td>North American Sales Manager</td>
<td>3</td>
</tr>
<tr>
<td>276</td>
<td>Linda Mitchell</td>
<td>Sales Representative</td>
<td>4</td>
</tr>
<tr>
<td>275</td>
<td>Michael Blythe</td>
<td>Sales Representative</td>
<td>4</td>
</tr>
<tr>
<td>285</td>
<td>Syed Abbas</td>
<td>Pacific Sales Manager</td>
<td>3</td>
</tr>
<tr>
<td>286</td>
<td>Lynn Tsoflias</td>
<td>Sales Representative</td>
<td>4</td>
</tr>
</tbody>
</table>
Other New Features

- RegEx Library changed
  - from Herny Spencer to ICU RegEx → Test!
- Internal Temporary Tables
  - TempTable SE replaces MEMORY SE, supports BLOB
- Error Logging was rewritten
- Backup Lock
  - Allows DML but prevents operations resulting in inconsistent snapshots.
    - LOCK INSTANCE FOR BACKUP
- Partial JSON update supported in Replication as well
- Special Admin connection
  - admin_address, admin_port (33062)
- Plug-ins must be (re-)written now in C++
Persistent Configuration

- (spfile)
- /var/lib/mysql/mysqld-auto.cnf (JSON)

SET PERSIST max_connections = 505;
Features Deprecated in 8.0

- Deprecated Feature: Removed in the future!

- utf8(mb3) → utf8mb4 (introduced in 5.5)
  - Not latin1!
- validate_password Plug-in → Component
- JSON_MERGE() → JSON_MERGE_PRESERVE()
- Various Tablespace functionality

Features Removed in 8.0

- Removed: Does NOT exist/work any more!

- `PASSWORD()`
  - → ... IDENTIFIED BY 'secret'
  - → ... IDENTIFIED WITH auth_plugin AS 'hash_string'

- InnoDB INFORMATION_SCHEMA views were renamed

- Query Cache (disabled since 5.6)

- `mysql_install_db` → `mysqld --initialize`

- Generic Partitioning (MyISAM)

- See Upgrade...

CREATE TABLE ptn_test (  
id INT UNSIGNED NOT NULL AUTO_INCREMENT  
, data VARCHAR(64)  
, ts TIMESTAMP  
, PRIMARY KEY (id, ts)  
)  
ENGINE = MyISAM  
PARTITION BY RANGE ( UNIX_TIMESTAMP(ts) ) (  
  PARTITION p_2018_01 VALUES LESS THAN  
  ( UNIX_TIMESTAMP('2018-02-01 00:00:00') )  
  ...  
, PARTITION p_2018_12 VALUES LESS THAN  
  ( UNIX_TIMESTAMP('2019-01-01 00:00:00') )  
, PARTITION p_max VALUES LESS THAN (MAXVALUE)  
);  
ERROR 1178 (42000): The storage engine for the table doesn't support native partitioning
Upgrade to MySQL 8.0

- RTFM!!!
- Client API is affected as well (libmysqlclient)!
- Check for Removed Features
  - Some Applications might not work any more!
- Test!!!
- Backup!
  - Downgrade is NOT supported!
- Upgrade from 5.6 to 8.0 is not supported!
- `mysqlcheck --all-databases --check-upgrade`
- `mysql_upgrade --user=root`
- Update can take very long time in some cases...
Questions?

Discussion?

We have time for some face-to-face talks...

- FromDual provides neutral and independent:
  - Consulting
  - Training
  - Remote-DBA
  - Support for MariaDB, Galera Cluster and MySQL