ORACLE

What's new in MySQL

(~2020 until 8.0.32)

Kenny Gryp
MySQL Product Manager

Kenny Gryp

MySQL Product Manager

ORACLE

MySQL 8.0

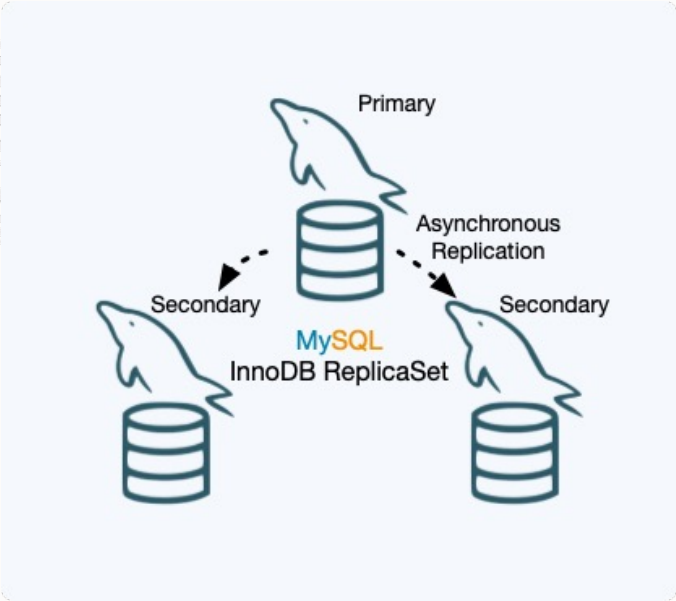
- Generally Available since April 2018
- Many bugfixes released since
- Continuous Development
  - Release new features much faster, in every release

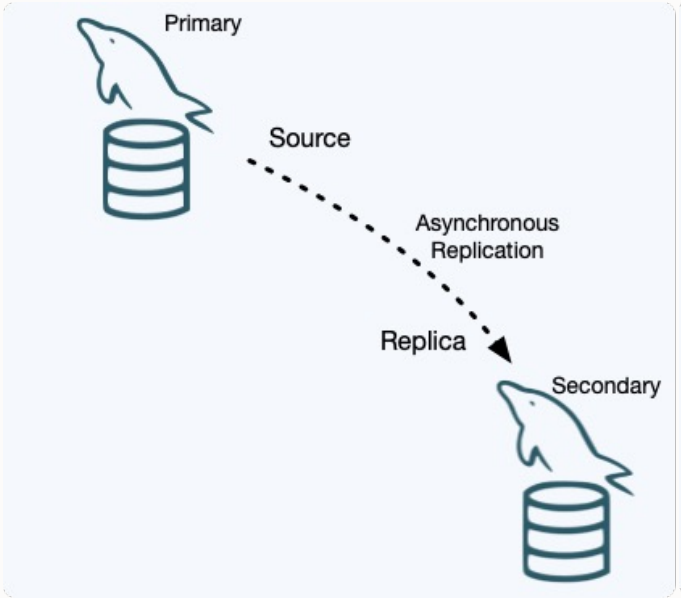
- 8.0 Community contributions
  - 714 contributions
  - 139 unique contributors

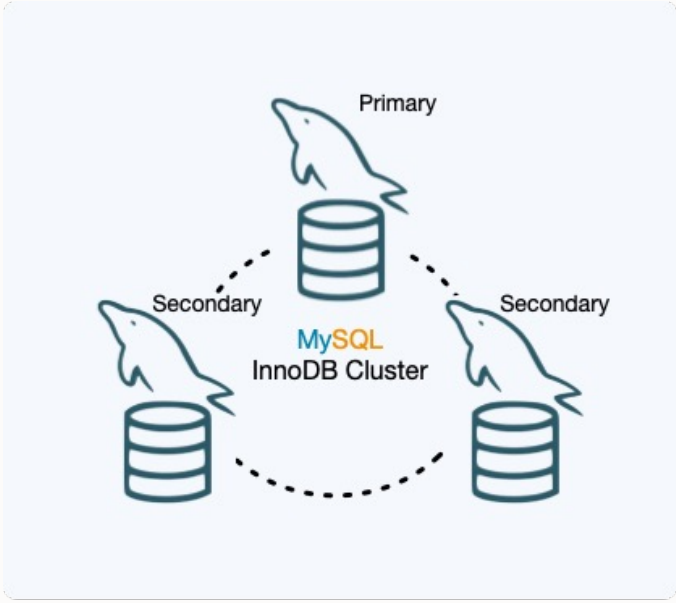
Let's dive into some of the features released introduced since 2020

PRIMARY / SECONDARY

- Remove offensive words
  - Not limited to only SQL Syntax e.g `SHOW REPLICA STATUS`
  - Including documentation, variables, error messages, code
- Work in progress
  - Backwards compatibility remains important
- Server roles:
  - `PRIMARY`,
  - `SECONDARY`,
  - `REPLICA`
- Replication configuration:
  - `SOURCE`,
  - `REPLICA`
- `ALLOWLIST`, `BLOCKLIST`







ORACLE

MySQL Server

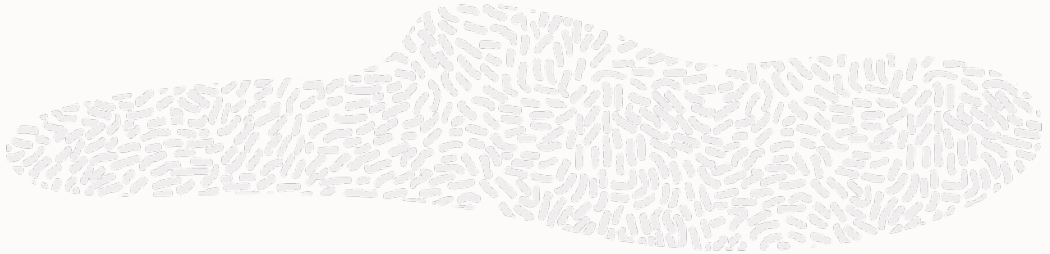
Administration

- Administrative Interface Support for Encrypted Connections – 8.0.21
  - `admin_tls_xxx` `admin_ssl_xxx`

- Error log through `performance_schema` – 8.0.22

```
mysql> SELECT LOGGED, ERROR_CODE, LEFT(DATA, 96) FROM performance_schema.error_log ORDER BY LOGGED DESC LIMIT 6;
+----------------------------+------------+------------------------------------------------------------------------------------------------+
| LOGGED                     | ERROR_CODE | LEFT(DATA, 96)                                                                                 |
+----------------------------+------------+------------------------------------------------------------------------------------------------+
| 2023-01-30 22:10:07.918430 | MY-011946  | Buffer pool(s) load completed at 230130 22:10:07                                               |
| 2023-01-30 22:10:07.837002 | MY-010931  | /usr/sbin/mysqld: ready for connections. Version: '8.0.32-commercial'  socket: '/var/lib/mysql/m |
| 2023-01-30 22:10:07.836941 | MY-011323  | X Plugin ready for connections. Bind-address: '::' port: 33060, socket: /var/run/mysqld/mysqlx.s |
| 2023-01-30 22:10:07.836844 | MY-011243  | Plugin mysqlx reported: 'Using OpenSSL for TLS connections'                                    |
| 2023-01-30 22:10:07.836400 | MY-011240  | Plugin mysqlx reported: 'Using SSL configuration from MySQL Server'                            |
| 2023-01-30 22:10:07.836218 | MY-010051  | Event Scheduler: scheduler thread started with id 5                                            |
+----------------------------+------------+------------------------------------------------------------------------------------------------+
6 rows in set (0.01 sec)
```

Accounts & Privileges



- Temporary Account Locking – 8.0.19

```
Mysql> CREATE USER 'u1'@'localhost’
         IDENTIFIED BY 'password’
         FAILED_LOGIN_ATTEMPTS 3
         PASSWORD_LOCK_TIME 3;
```

- User Attributes & Comments – 8.0.21

```
mysql> CREATE USER 'bill'@'localhost’
         COMMENT 'This is Bill\'s user account’;
mysql> ALTER USER 'mary'@'localhost’
         ATTRIBUTE '{"email":"bill@example.com"}';
```


Generated Invisible Primary Keys – 8.0.30

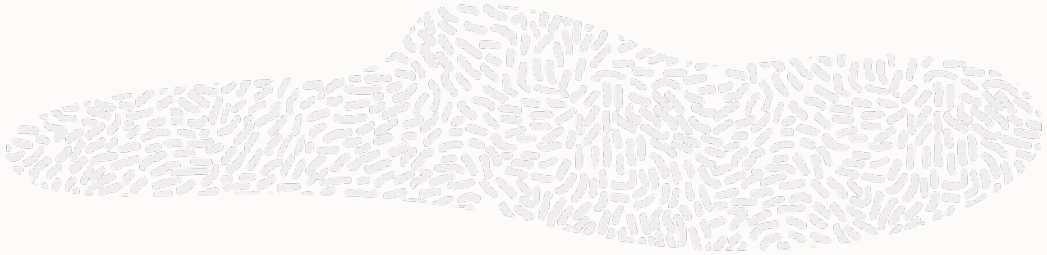
- INVISIBLE columns – 8.0.23
- Automatically add PK to tables without PK – 8.0.30
- Replication Support – 8.0.31
  - Source without PK
  - Replica adds PK

```
CHANGE REPLICATION SOURCE
REQUIRE_TABLE_PRIMARY_KEY_CHECK=GENERATE
```

- Solves:
  - MySQL Group Replication Requirement of PK
  - Replication applier performance
  - Remove scalability issues with `dict_sys`

```
mysql> CREATE TABLE lefred (tw INT, at INT);
mysql> SHOW CREATE TABLE lefred\G
*************************** 1. row ***************************
       Table: lefred
Create Table: CREATE TABLE `lefred` (
  `tw` int DEFAULT NULL,
  `at` int DEFAULT NULL
) ENGINE=InnoDB


mysql> SET SESSION sql_generate_invisible_primary_key=on;
mysql> CREATE TABLE miguel (tw INT, at INT);
mysql> SHOW CREATE TABLE miguel\G
*************************** 1. row ***************************
       Table: miguel
Create Table: CREATE TABLE `miguel` (
  `my_row_id` bigint unsigned NOT NULL AUTO_INCREMENT INVISIBLE,
  `tw` int DEFAULT NULL,
  `at` int DEFAULT NULL,
  PRIMARY KEY (`my_row_id`)
) ENGINE=InnoDB
```

Optimizer



- User defined histograms – 8.0.31
  - Manually specify histogram values
  - Offload histogram to different servers

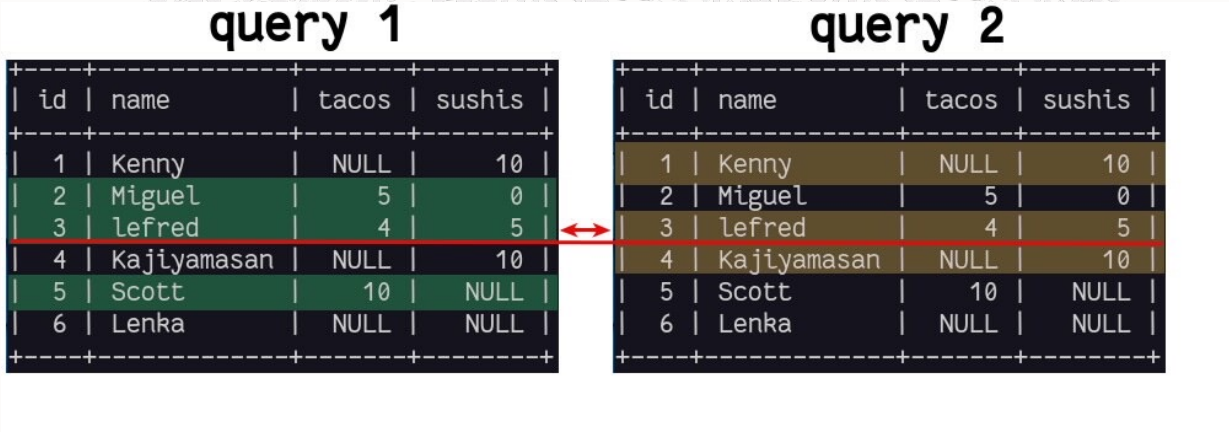
```
ANALYZE TABLE t UPDATE HISTOGRAM ON c1 USING DATA '<JSON>'
```

- New Index Hints for `FORCE INDEX` and `IGNORE INDEX` – 8.0.20

- Prepared Statements: Prepare each DML statement once – 8.0.22

Optimizer

- INTERSECT, EXCEPT – 8.0.31
- JSON_VALUE – 8.0.21
- CHECK constraints – 8.0.16
- JSON Validation – 8.0.17
  - Also available in DocStore – 8.0.21




```
select * from new where tacos > 0
intersect
select * from new where sushis > 0;
+----+--------+-------+--------+
| id | name   | tacos | sushis |
+----+--------+-------+--------+
|  3 | lefred |     4 |      5 |
+----+--------+-------+--------+
```

Optimizer

- Memory tracking and limiting - 8.0.28
  - `global_connection_memory_limit` - Total amount of memory allowed
    `connection_memory_limit` - Limit how much memory a single connection can use
- `HASH JOINS` – 8.0.18
- `ORDER|GROUP BY … LIMIT..`
  - `optimizer_switch=prefer_ordering_index=off` – 8.0.21
- `EXPLAIN ANALYZE` – 8.0.18

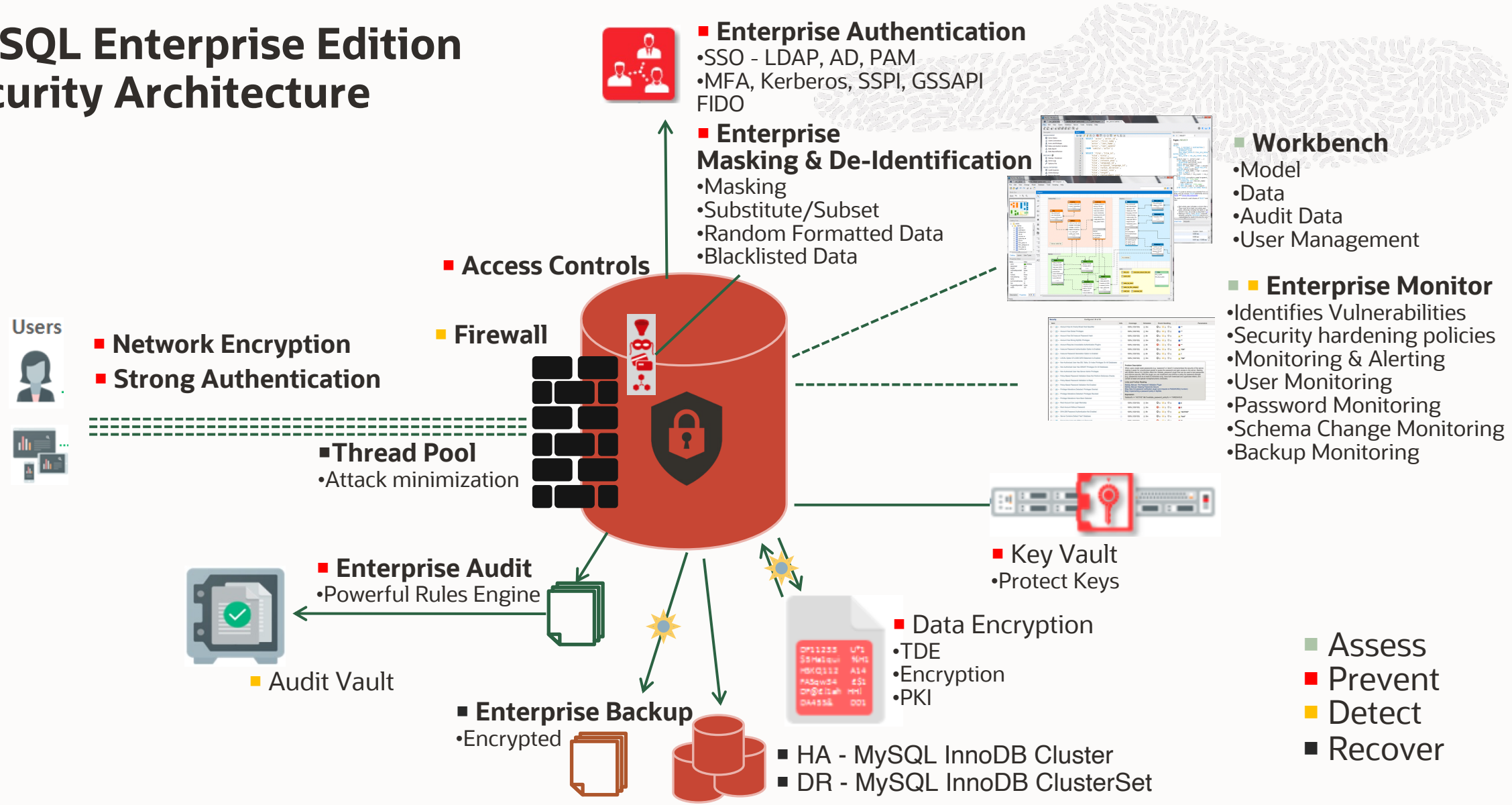
```
EXPLAIN ANALYZE
SELECT first_name, last_name, SUM(amount) AS total
FROM staff INNER JOIN payment
  ON staff.staff_id = payment.staff_id
     AND
     payment_date LIKE '2005-08%'
GROUP BY first_name, last_name;

-> Table scan on <temporary>  (actual time=0.001..0.001 rows=2 loops=1)
    -> Aggregate using temporary table  (actual time=58.104..58.104 rows=2 loops=1)
        -> Nested loop inner join  (cost=1757.30 rows=1787) (actual time=0.816..46.135 rows=5687 loops=1)
            -> Table scan on staff  (cost=3.20 rows=2) (actual time=0.047..0.051 rows=2 loops=1)
            -> Filter: (payment.payment_date like '2005-08%')  (cost=117.43 rows=894) (actual time=0.464..22.767 rows=2844 loops=2)
                -> Index lookup on payment using idx_fk_staff_id (staff_id=staff.staff_id)  (cost=117.43 rows=8043) (actual time=0.450..19.988
rows=8024 loops=2)
```

GIS Improvements

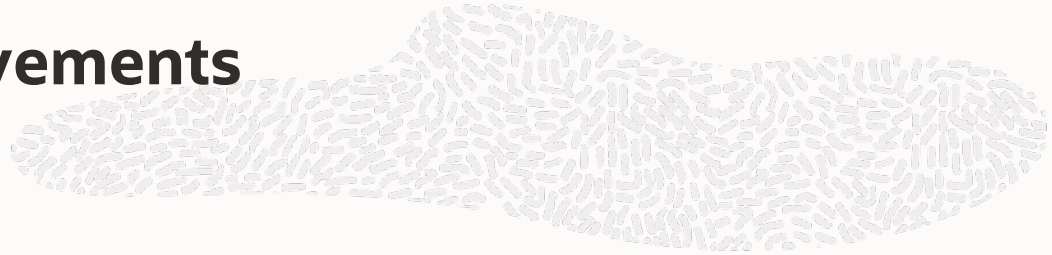
- Spatial relations
  - `ST_HausdorffDistance`
  - `ST_FrechetDistance`
- Spatial Operators
  - `ST_LineInterpolatePoint`
  - `ST_LineInterpolatePoints`
  - `ST_PointAtDistance`
- Spatial aggregations
  - `ST_Collect`
- Cast between geometry types
- `ST_Transform`: Popular Visualisation Pseudo Mercator (EPSG 1024) & Projections

- Geographic (long-lat) support where we previously only supported Cartesian (flat map):
  - `ST_Distance`
  - `ST_Buffer`
  - `ST_Difference`
  - `ST_Union`
  - `ST_SymDifference`
  - `ST_Intersection`

MySQL Enterprise Edition Security Architecture



MySQL Enterprise Authentication Improvements

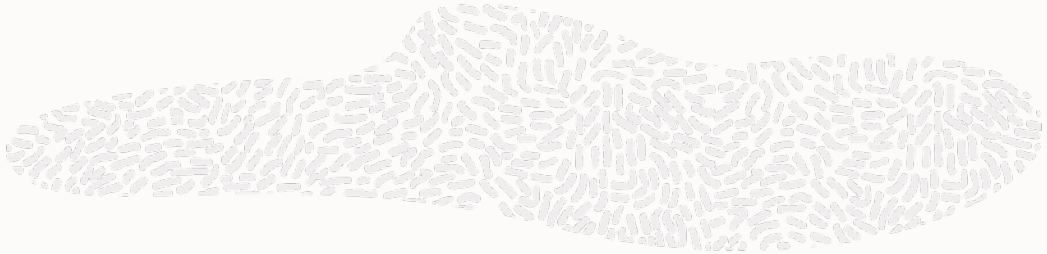


- Native Kerberos (SSPI, GSSAPI) – 8.0.26
- Multifactor Authentication (MFA) - 8.0.27
- Password-less Authentication - FIDO (Fast Identity Online) – 8.0.27

ORACLE

InnoDB

InnoDB Doublewrite



- More flexibility – 8.0.20
  - Choose Directory
  - Define how many files
  - Control flushing
- Encrypted Doublewrite – 8.0.23
- More performance: `DETECT_ONLY` – 8.0.30

InnoDB – Redo Log

- Disable Redo Logging – 8.0.21
  - Less write amplification on Replicas
  - Much faster logical import (MySQL Shell Dump & Load
  - (Also disables doublewrite)
  - (Be careful doing this in production)

```
mysql> ALTER INSTANCE DISABLE INNODB REDO_LOG;
Query OK, 0 rows affected (0.00 sec)


mysql> ALTER INSTANCE ENABLE INNODB REDO_LOG;
Query OK, 0 rows affected (0.01 sec)
```

- Reconfigure REDO LOG size online – 8.0.30

```
mysql> SHOW STATUS LIKE 'Innodb_redo_log_capacity_resized';
+-----------------------------------+-------------+
| Variable_name                     | Value       |
+-----------------------------------+-------------+
| Innodb_redo_log_capacity_resized | 137438953472 |
+-----------------------------------+-------------+
1 row in set (0.00 sec)

mysql> SET GLOBAL innodb_redo_log_capacity = 8589934592;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW STATUS LIKE 'Innodb_redo_log_capacity_resized';
+-----------------------------------+------------+
| Variable_name                     | Value      |
+-----------------------------------+------------+
| Innodb_redo_log_capacity_resized | 8589934592 |
+-----------------------------------+------------+
1 row in set (0.00 sec)
```

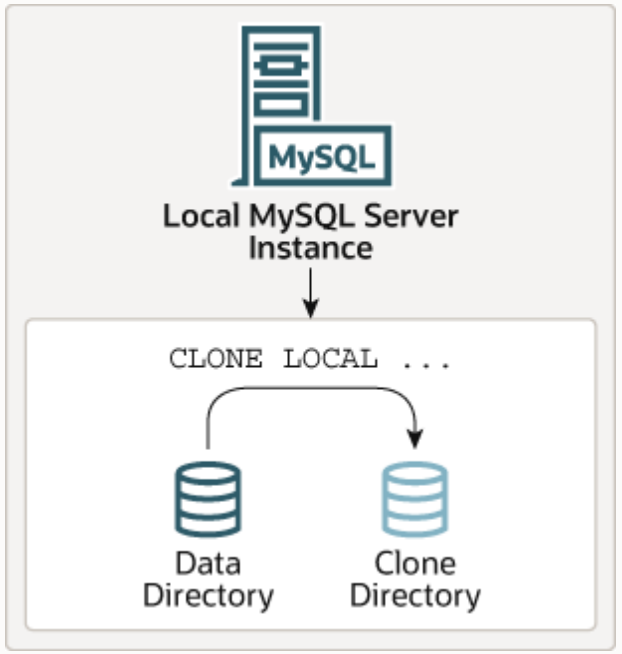
InnoDB – Instant DDL

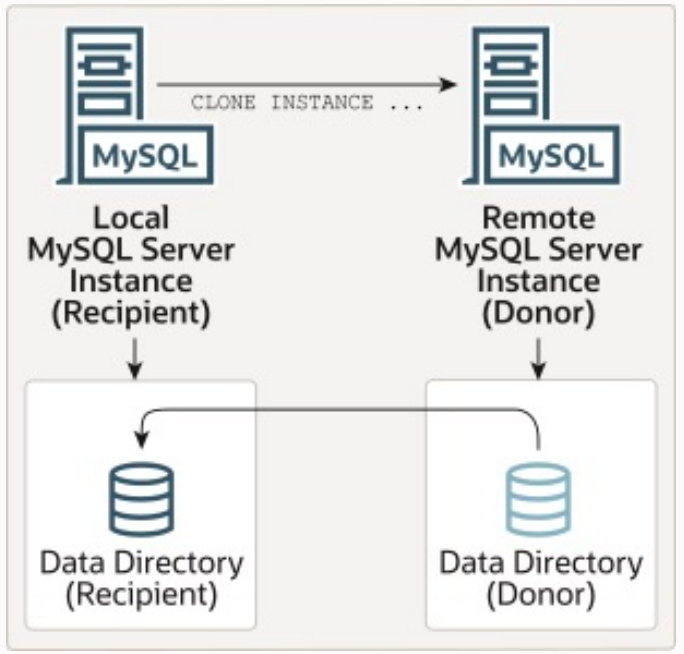
- RENAME COLUMN – 8.0.28
- ADD/DROP now INSTANT (everywhere) – 8.0.29

```
mysql> SELECT COUNT(*) FROM miguel;
+----------+
| count(*) |
+----------+
|  1000000 |
+----------+

mysql> SHOW CREATE TABLE miguel\G
*************************** 1. row ***************************
       Table: miguel
Create Table: CREATE TABLE `miguel` (
  `id` int NOT NULL AUTO_INCREMENT,
  `k` int NOT NULL DEFAULT '0',
  `c` char(120) NOT NULL DEFAULT '',
  `pad` char(60) NOT NULL DEFAULT '',
  PRIMARY KEY (`id`),
  KEY `k_1` (`k`)
) ENGINE=InnoDB AUTO_INCREMENT=1000001

mysql> ALTER TABLE miguel ADD COLUMN lefred int unsigned AFTER k;
Query OK, 0 rows affected (0.04 sec)

mysql> ALTER TABLE miguel DROP COLUMN c;
Query OK, 0 rows affected (0.03 sec)

mysql> SELECT NAME, TOTAL_ROW_VERSIONS FROM
INFORMATION_SCHEMA.INNODB_TABLES WHERE NAME LIKE 'daniel/miguel';
+----------------+----------------------+
| NAME           | TOTAL_ROW_VERSIONS   |
+----------------+----------------------+
| daniel/miguel  |                    2 |
+----------------+----------------------+
```

InnoDB – CLONE – 8.0.17

- Create physical snapshot of server
  - Local
  - Remote
- Using MySQL Classic port & MySQL Authentication
- Integrated in MySQL InnoDB ReplicaSet/Cluster/ClusterSet

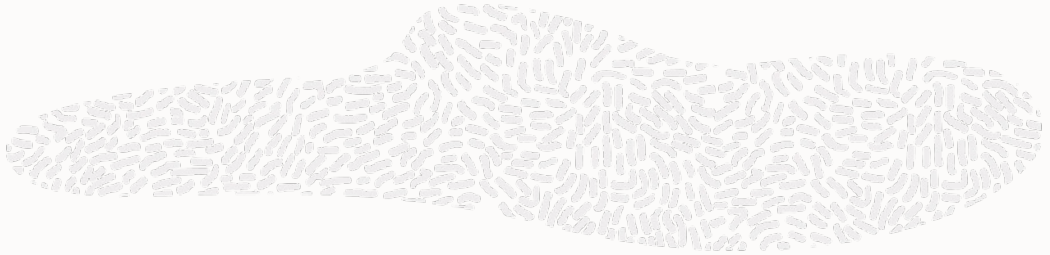




InnoDB Parallel Index Creation of Secondary Indexes

- Creating Secondary index involves:
  1. Scanning the clustered index
     - Parallelized in 8.0.27:
       - `innodb_parallel_read_threads` (global)
     - Depends on # subtrees
  2. Sorting the data
     - `innodb_ddl_threads` – 8.0.27
  3. Loading into the secondary index
     - Not yet parallelized

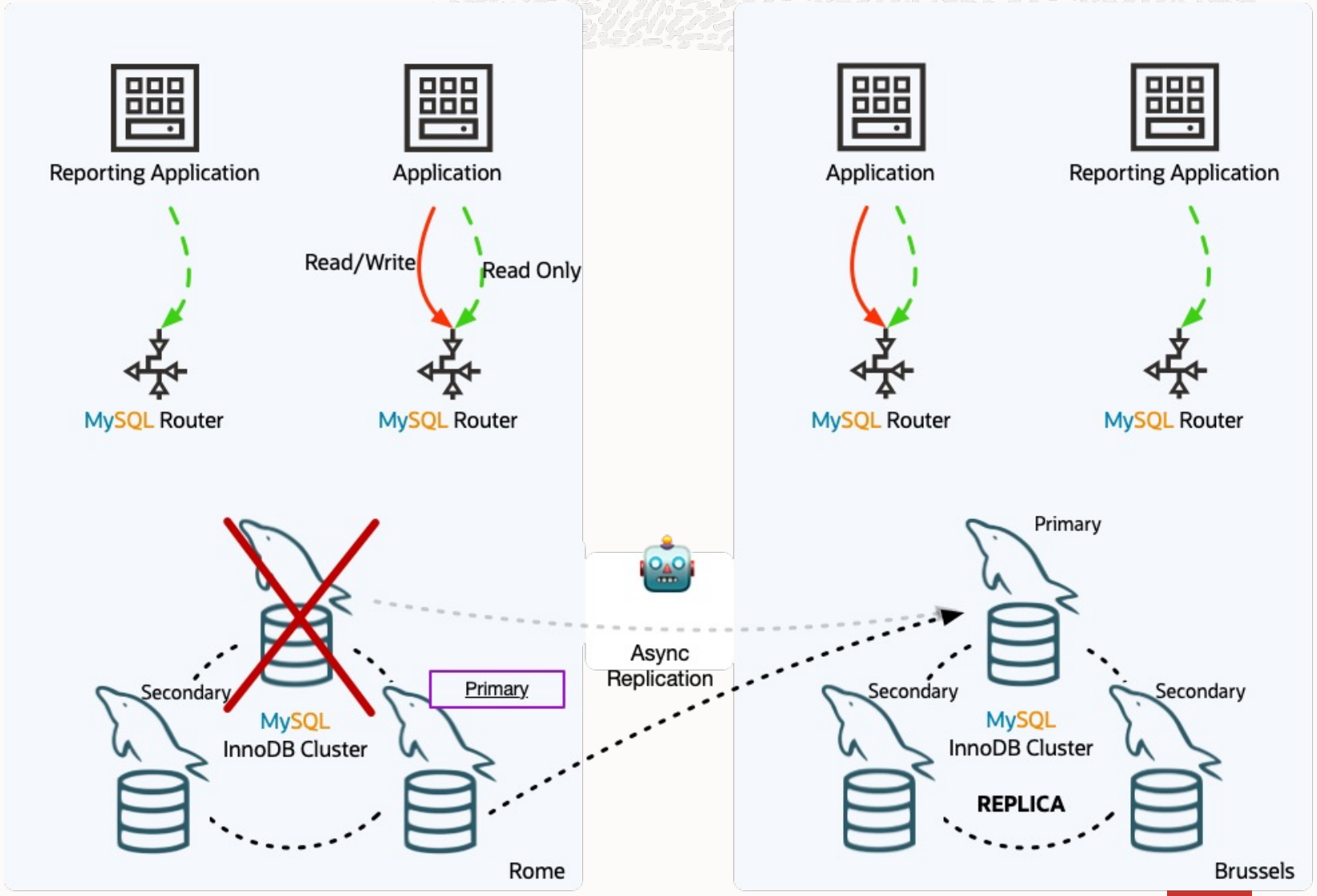
Replication & Group Replication



- Binlog transaction compression – 8.0.20
  - `zstd`
  - Compressed binlog events
  - Saves on disk space & network traffic
  - Compressed event contains entire trx, can improve applier performance
- Replication From a Source Without GTIDs to a Replica With GTIDs – 8.0.23
  - Alternative migration into GTID
  - `CHANGE REPLICATION SOURCE … ASSIGN_GTIDS_TO_ANONYMOUS_TRANSACTIONS = {OFF | LOCAL | uuid}`
- MySQL Authentication for internal communication (XCOM) – 8.0.27
  - Secure and authenticated connections
- XCOM: PAXOS Single Leader mode – 8.0.27
  - Improves throughput/latency in single primary mode

Automatic Asynchronous Replication Connection Failover – 8.0.22

- Define set of servers as source – 8.0.22
- Use MySQL Group Replication topology as source – 8.0.23
  - Prefer `PRIMARY` or `SECONDARY`
  - MySQL InnoDB ClusterSet: Automates topology changes in Disaster Recovery deployments



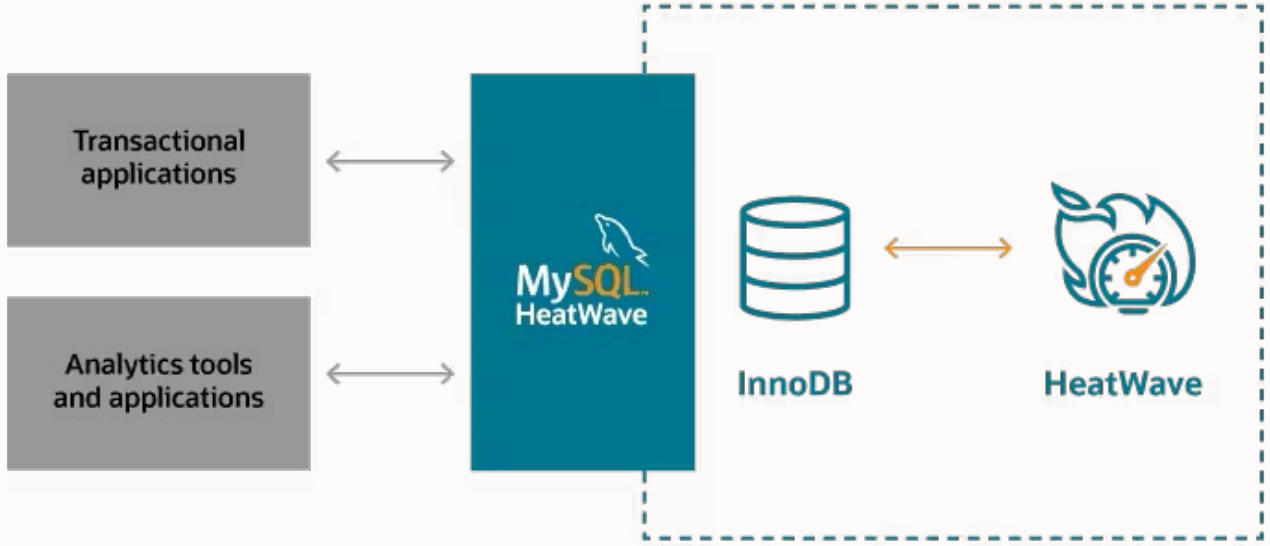
ORACLE

MySQL HeatWave for OLAP

MySQL Heatwave for OLAP

HeatWave—In-Memory Query Accelerator with Built-in ML

- Increases MySQL performance by orders of magnitude for analytics and mixed workloads. Eliminates the need for a separate analytics database, separate machine learning (ML) tools, and extract, transform, and load (ETL) duplication. MySQL HeatWave is available on Oracle Cloud Infrastructure (OCI), Amazon Web Services (AWS), and Microsoft Azure.



https://www.oracle.com/mysql/heatwave/

ORACLE CLOUD Infrastructure

aws

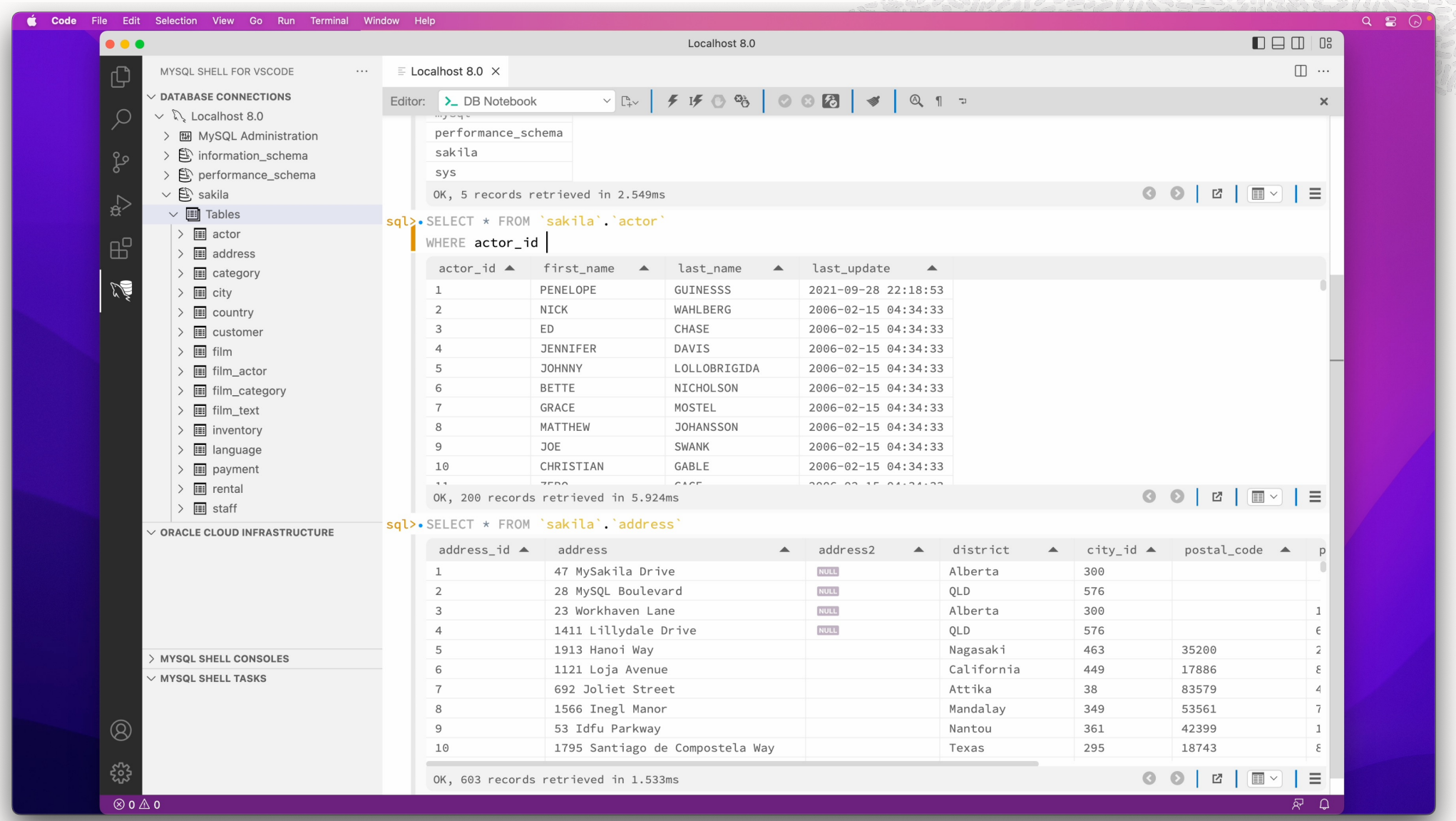
ORACLE

MySQL Shell

MySQL Shell

- Login options (`my.cnf` and `--defaults-file`) – 8.0.32
- MySQL Shell `py` & `js` plugins – 8.0.17
- SSH Tunneling – 8.0.28
- Diagnostics Collection
  - `util.debug.collectDiagnostics` – 8.0.29
  - `util.debug.collectSlowQueryDiagnostics` – 8.0.31
  - `util.debug.collectHighLoadDiagnostics` – 8.0.31

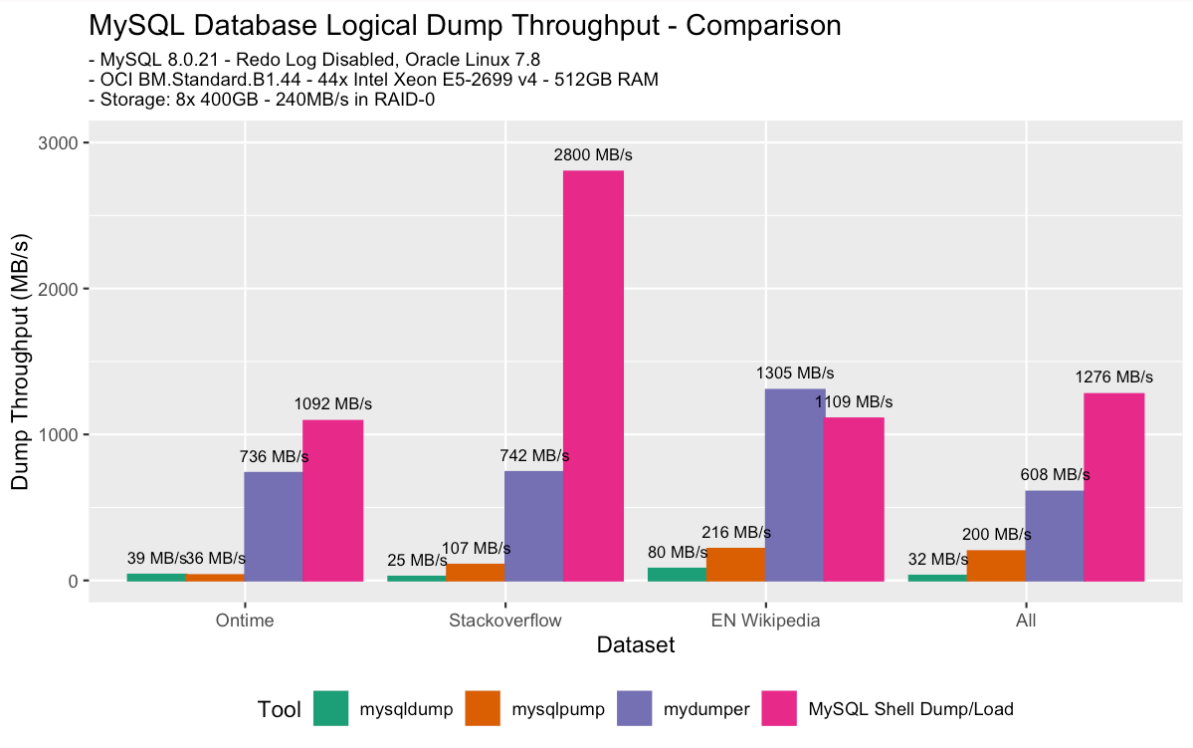
MySQL Shell Plugin for VSCode

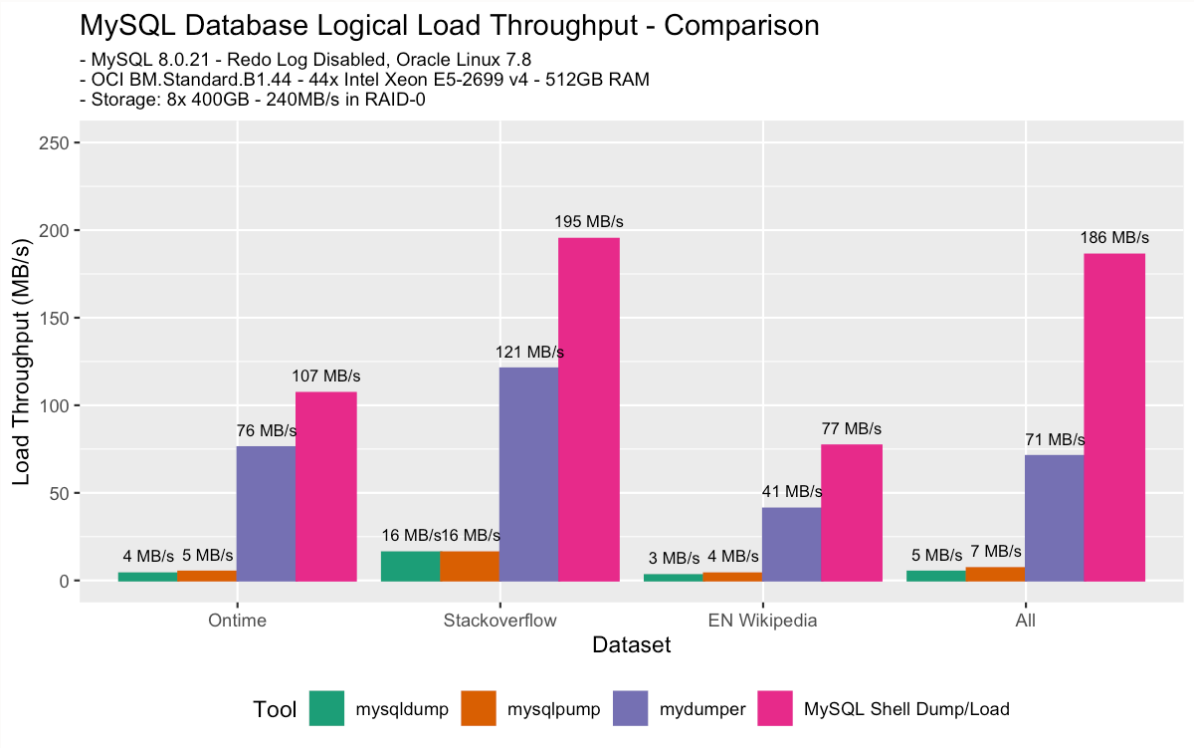


MySQL Shell Dump & Load – 8.0.21

- Logical dump/load
- OCI Object Storage, AWS S3, Azure Blob
- Load while dump
- Load: stop and resume

- Very fast: multi-threaded dump/load
  - `ALTER INSTANCE DISABLE INNODB REDO_LOG`
  - `innodb-extend-and-initialize=0`



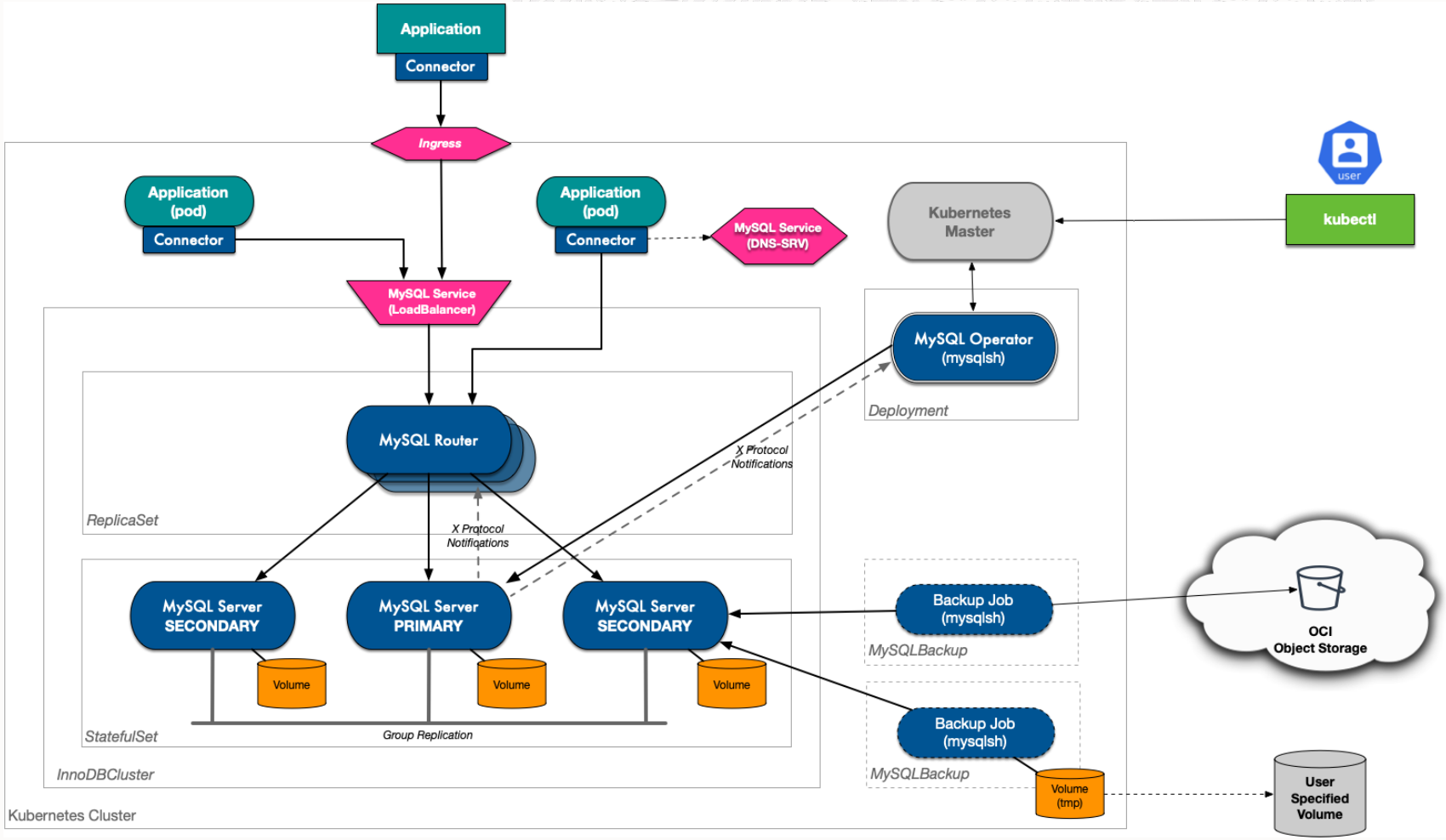


ORACLE

MySQL Operator for Kubernetes

MySQL Operator for Kubernetes

- Automated deployment and management of
  - MySQL Server
  - MySQL InnoDB Cluster
  - MySQL Router
- Self-healing
- Backup & Restore
- Scaleup/Scaledown of Router & Server
- Rolling upgrades with minimal downtime
- Configuration Management
- Deploy from InnoDB CLONE
- Private container registries



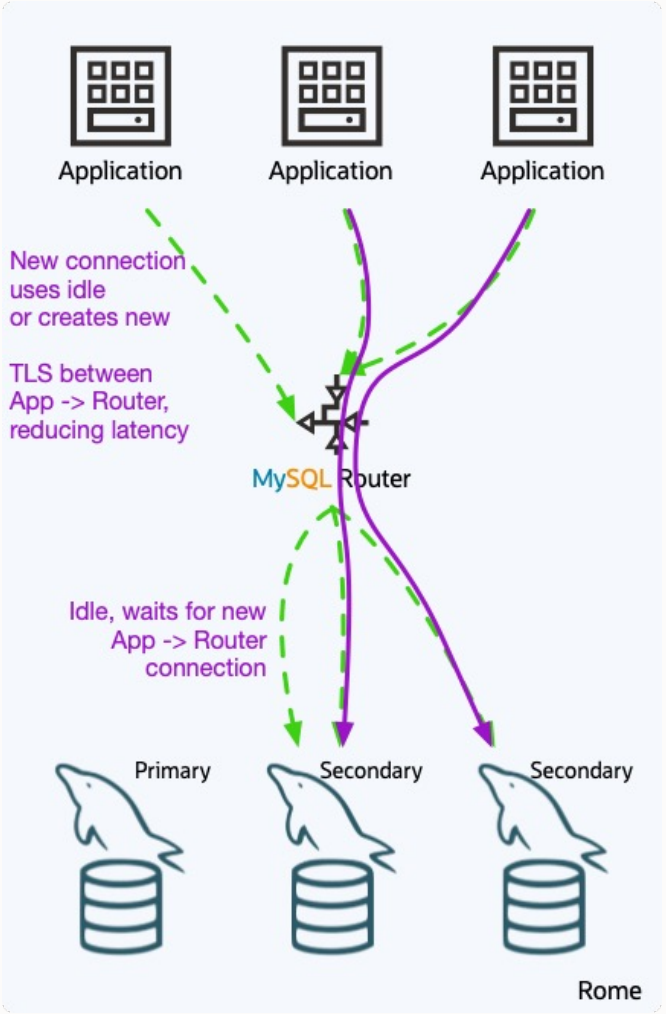
https://github.com/mysql/mysql-operator

ORACLE

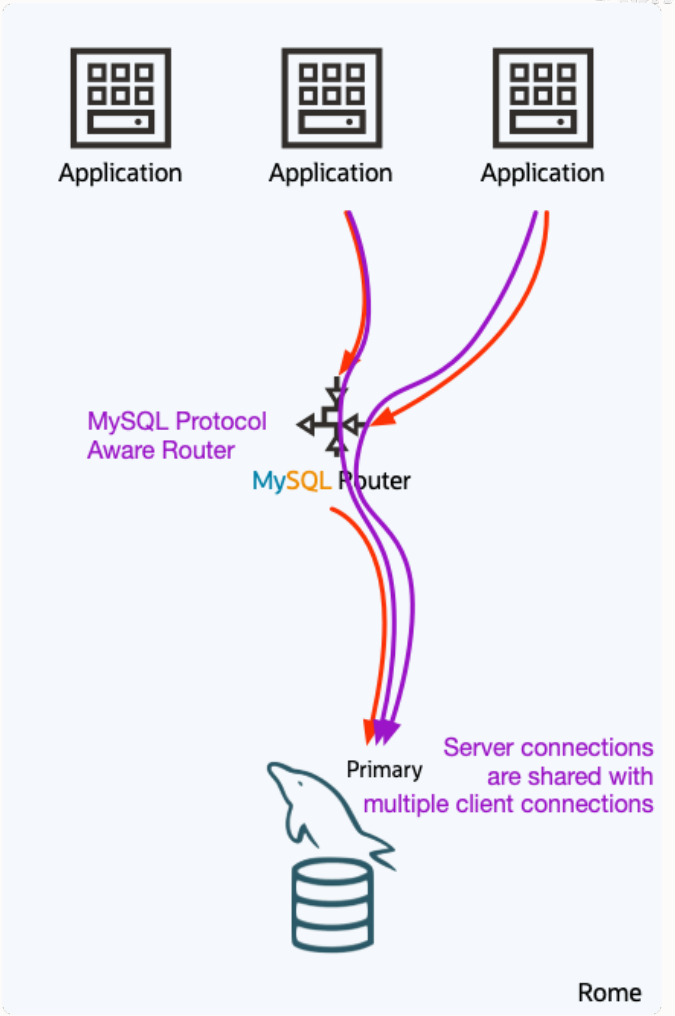
MySQL Router

MySQL Router

Connection Reuse & TLS Offload – 8.0.29



Connection Multiplexing – 8.0.32

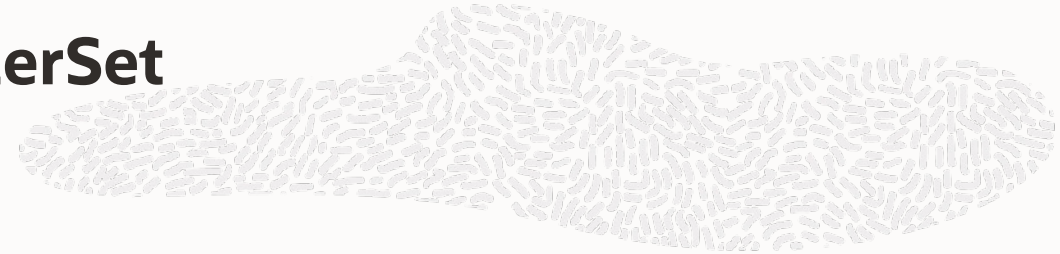


ORACLE

MySQL InnoDB ReplicaSet, Cluster & ClusterSet

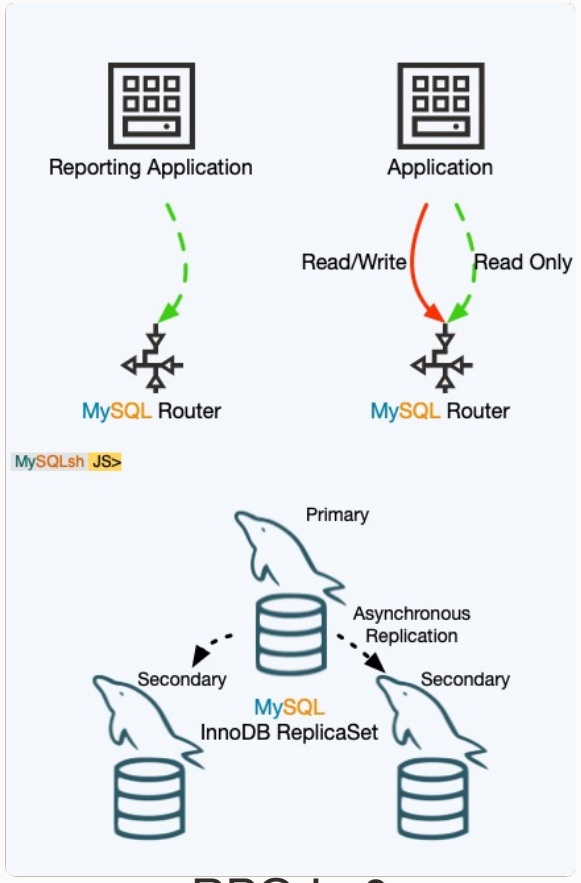
MySQL InnoDB ReplicaSet/Cluster/ClusterSet

Move along into the present people…



- Are you stuck in the past?
  - Setting up Replication topology was usually done manually, taking many steps
    - Including user management, restoring backups, configuring replication…
  - MySQL only offered the technical pieces, leaving it up to the user to setup an (always customized) architecture
  - Even required other software … bringing lot's of work for DBA's and experts, who spent their time automating and integrating their customized architecture

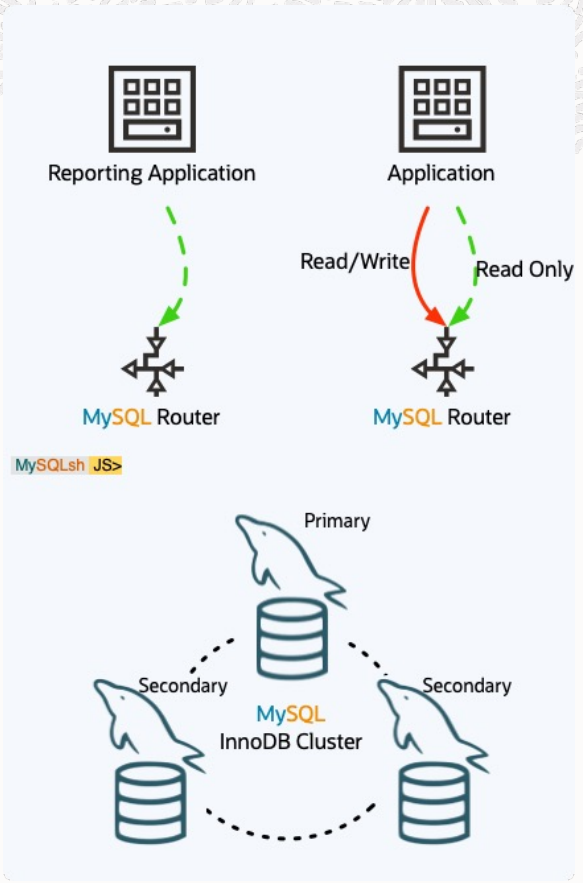
MySQL InnoDB ReplicaSet



RPO != 0

RTO = minutes (manual failover)

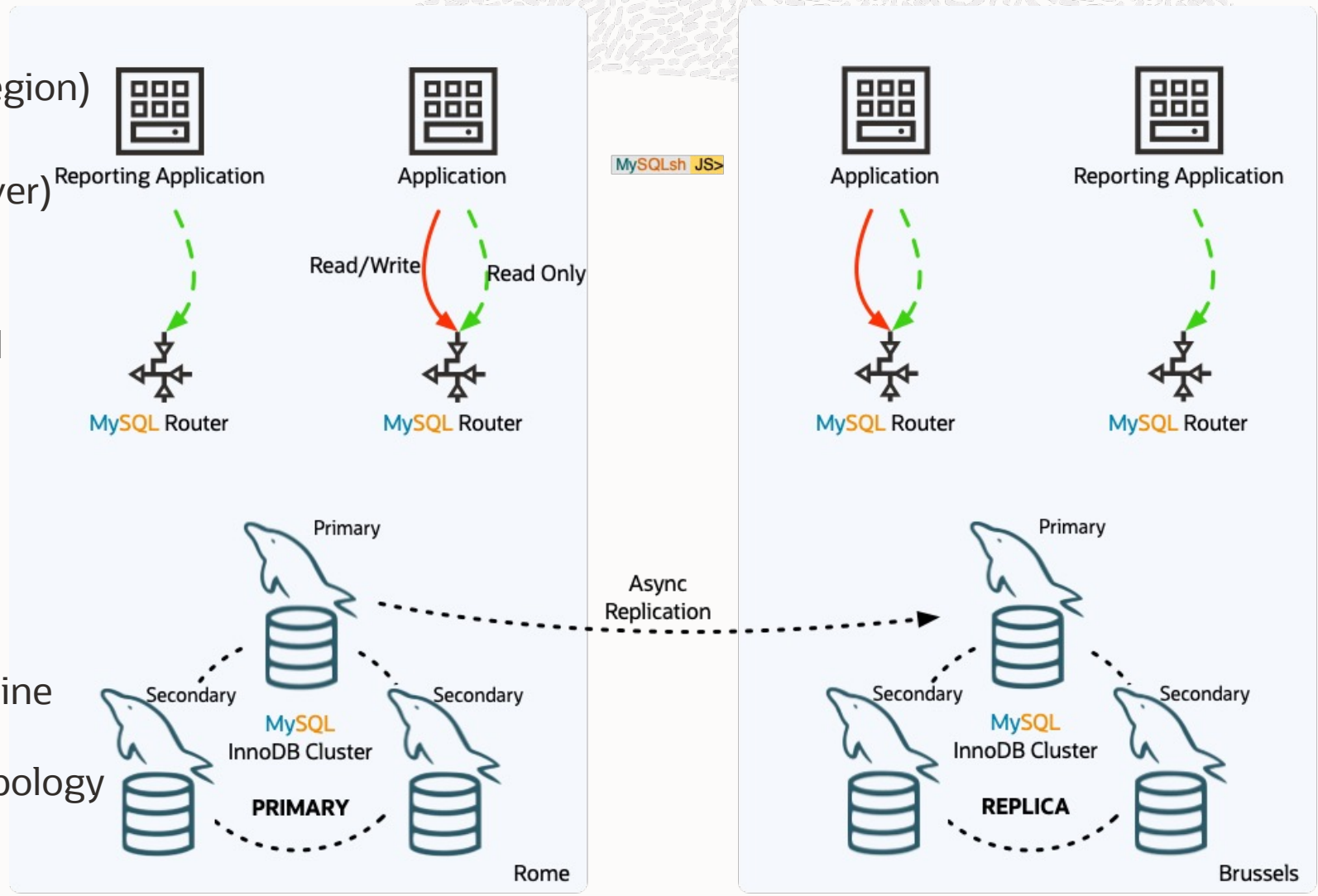
MySQL InnoDB Cluster



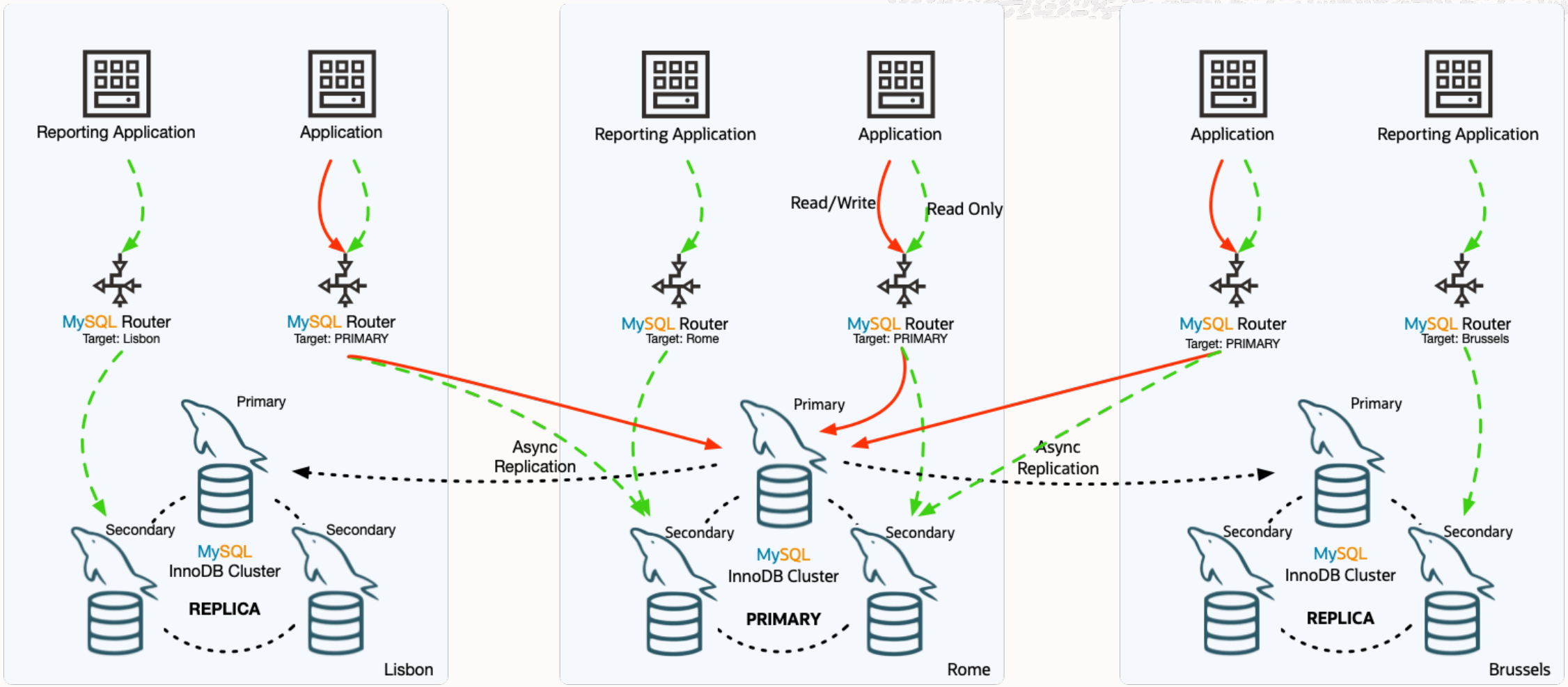
- RPO = 0
- RTO = seconds (automatic failover)

Disaster Recovery - MySQL InnoDB ClusterSet - 8.0.27

- High Availability (Failure Within a Region)
  - RPO = 0
  - RTO = seconds (automatic failover)
- Disaster Recovery (Region Failure)
  - RPO != 0
  - RTO = minutes or more (manual failover)
  - No write performance impact
- Features
  - Easy to use
  - Familiar interface and usability
  - `mysqlsh`, `CLONE`, ...
  - Add/remove nodes/clusters online
  - Router integration, no need to reconfigure application if the topology changes



Disaster Recovery - MySQL InnoDB ClusterSet - 8.0.27




ORACLE