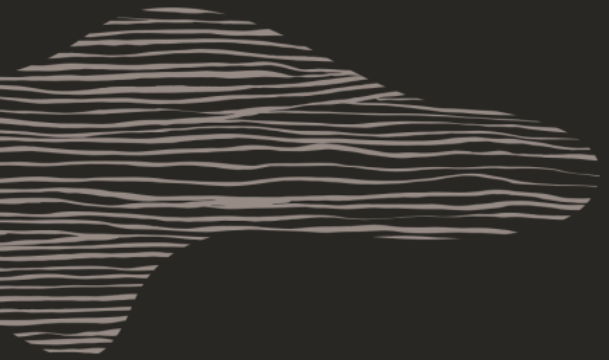
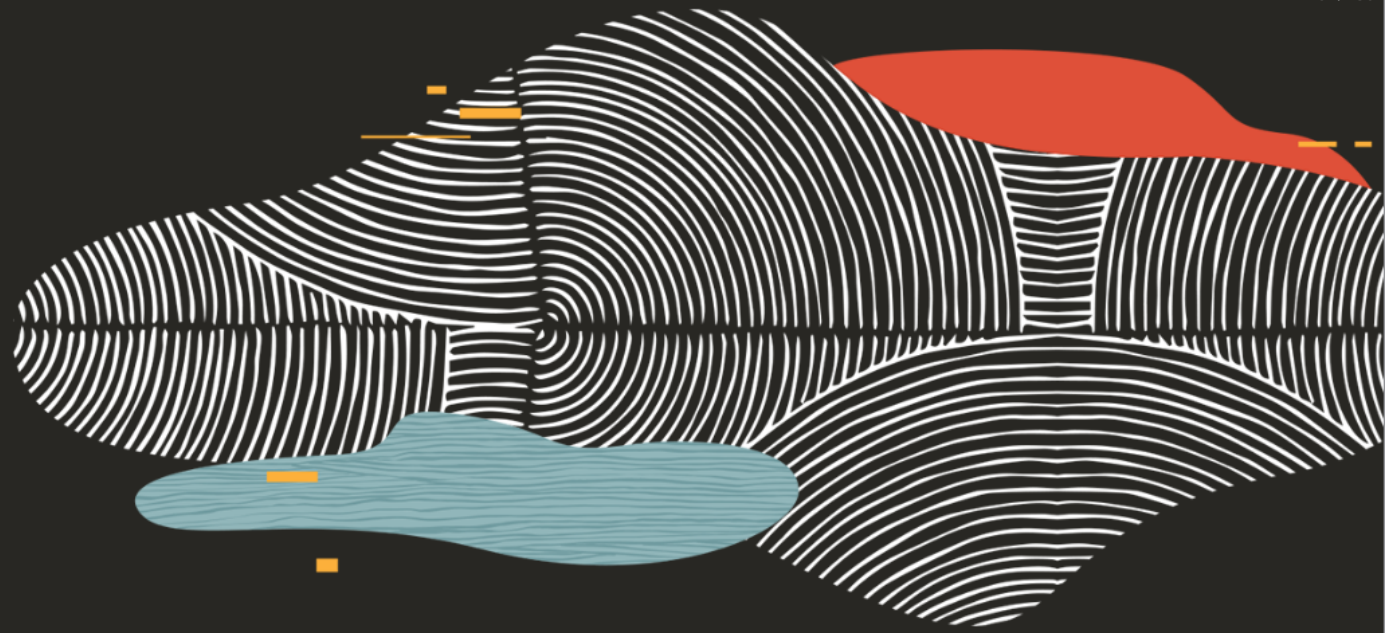
InnoDB Cluster

Now including Best Practices!

Kenny Gryp
MySQL Product Manager


ORACLE®





Safe Harbor Statement

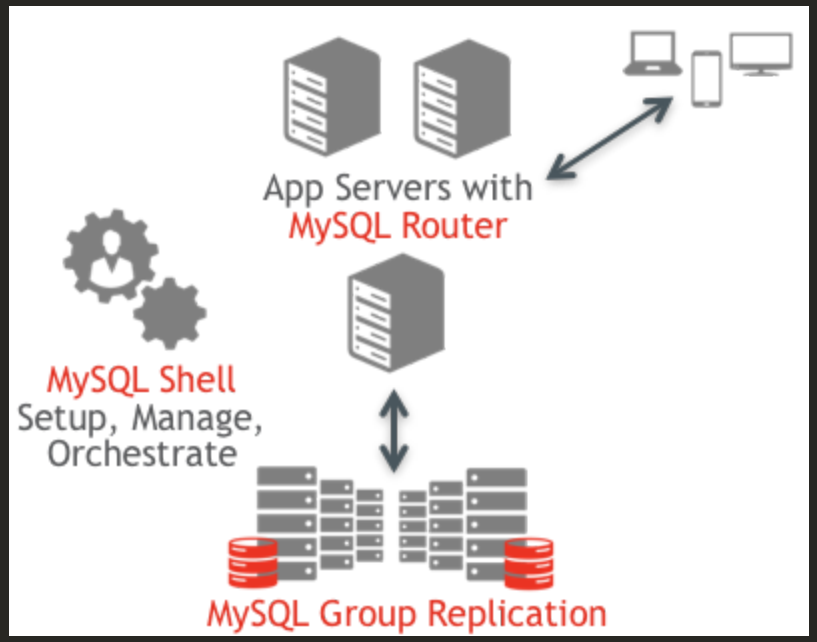
The following is intended to outline our general product direction. It is intended for information purpose only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied up in making purchasing decisions. The development, release and timing of any features or functionality described for Oracle´s product remains at the sole discretion of Oracle.

MySQL InnoDB Cluster

"A single product — MySQL — with **high availability** and **scaling features** baked in;
providing an **integrated end-to-end solution that is easy to use**."

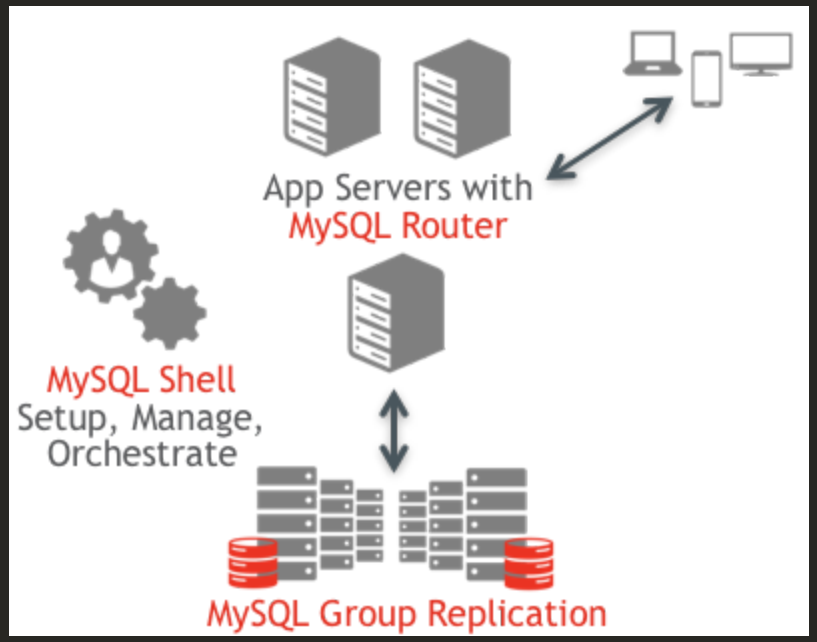
MySQL InnoDB Cluster

"A single product — MySQL — with **high availability** and **scaling features** baked in;
providing an **integrated end-to-end solution that is easy to use**."



MySQL InnoDB Cluster

"A single product — MySQL — with **high availability** and **scaling features** baked in; providing an **integrated end-to-end solution that is easy to use**."



Components:

- MySQL Group Replication
- MySQL Shell
- MySQL Router

MySQL InnoDB Cluster - Goals

One Product: MySQL

- All components developed together
- Integration of all components
- Full stack testing

MySQL InnoDB Cluster - Goals

One Product: MySQL

- All components developed together
- Integration of all components
- Full stack testing

Easy to Use

- One client: MySQL Shell
- Integrated orchestration
- Homogenous servers

MySQL InnoDB Cluster - Goals

One Product: MySQL

- All components developed together
- Integration of all components
- Full stack testing

Easy to Use

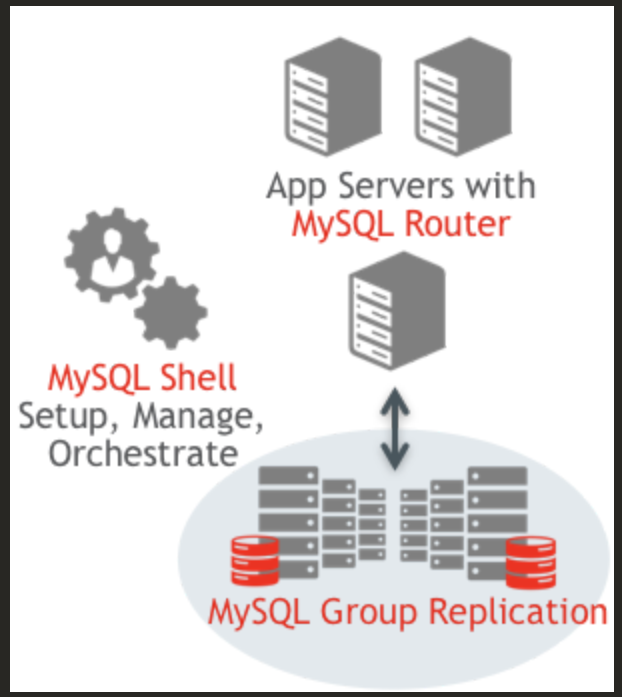
- One client: MySQL Shell
- Integrated orchestration
- Homogenous servers

Flexible and Modern - X Protocol

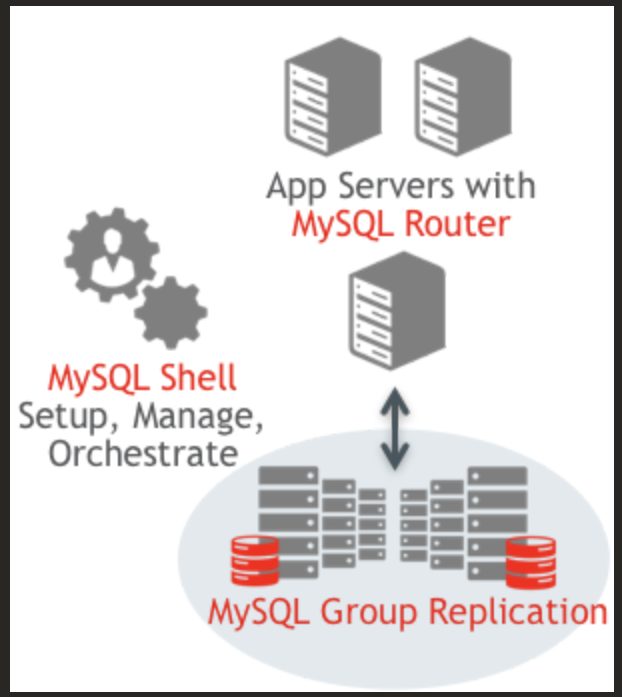
- SQL and NoSQL together
- Protocol Buffers
- Asynchronous API
- Developer friendly

MySQL Group Replication

MySQL Group Replication



MySQL Group Replication



High Available Distributed MySQL DB

- Fault tolerance
- Automatic failover
- Active/Active update anywhere (limits apply)
- Automatic membership management
  - Adding/removing members
  - Network partitions, failures
- Conflict detection and resolution
- Prevents data loss

MySQL Group Replication

- Implementation of Replicated Database State Machine
  - Total Order - Writes
  - XCOM - Paxos implementation
- Configurable Consistency Guarantees
  - eventual consistency
  - 8.0+: per session & global read/write consistency
- Using MySQL replication framework by design
  - binary logs
  - relay logs
  - GTIDs: Global Transaction IDs
- Generally Available since MySQL 5.7
- Supported on all platforms: linux, windows, solaris, macosx, freebsd

MySQL Group Replication - Use Cases

Consistency: No Data Loss (RPO=0)

- in event of failure of (primary) member
- Split brain prevention (Quorum)

MySQL Group Replication - Use Cases

Consistency: No Data Loss (RPO=0)

- in event of failure of (primary) member
- Split brain prevention (Quorum)

Highly Available: Automatic Failover

- Primary members are automatically elected
- Automatic Network Partition handling

MySQL Group Replication - Use Cases

Consistency: No Data Loss (RPO=0)

- in event of failure of (primary) member
- Split brain prevention (Quorum)

Highly Available: Automatic Failover

- Primary members are automatically elected
- Automatic Network Partition handling

Read Scaleout

- Add/Remove members as needed
- Replication Lag handling with Flow Control
- Configurable Consistency Levels
  - Eventual
  - Full Consistency -- no stale reads

MySQL Group Replication - Use Cases

Consistency: No Data Loss (RPO=0)

- in event of failure of (primary) member
- Split brain prevention (Quorum)

Highly Available: Automatic Failover

- Primary members are automatically elected
- Automatic Network Partition handling

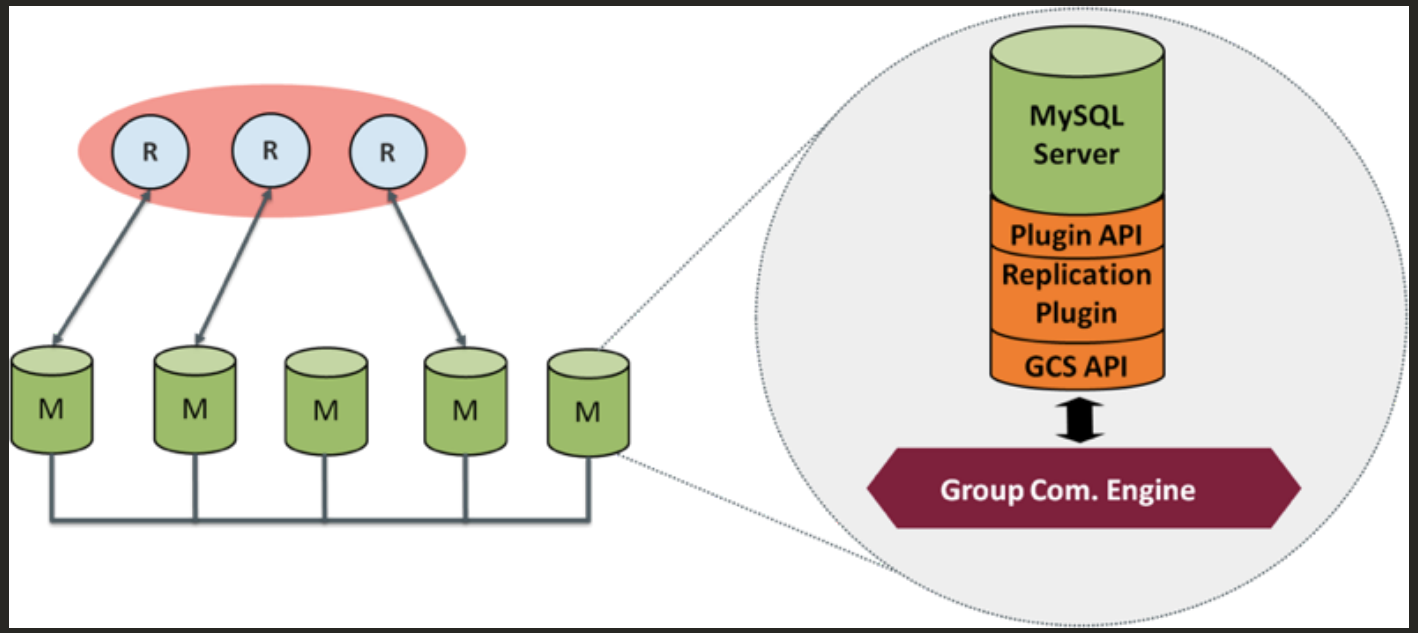
Read Scaleout

- Add/Remove members as needed
- Replication Lag handling with Flow Control
- Configurable Consistency Levels
  - Eventual
  - Full Consistency -- no stale reads

Active/Active environments

- Write to many members at the same time
  - ordered writes within the group (XCOM)
  - guaranteed consistency
- Good write performance
  - due to Optimistic Locking
    (workload dependent)

MySQL Group Replication - Architecture

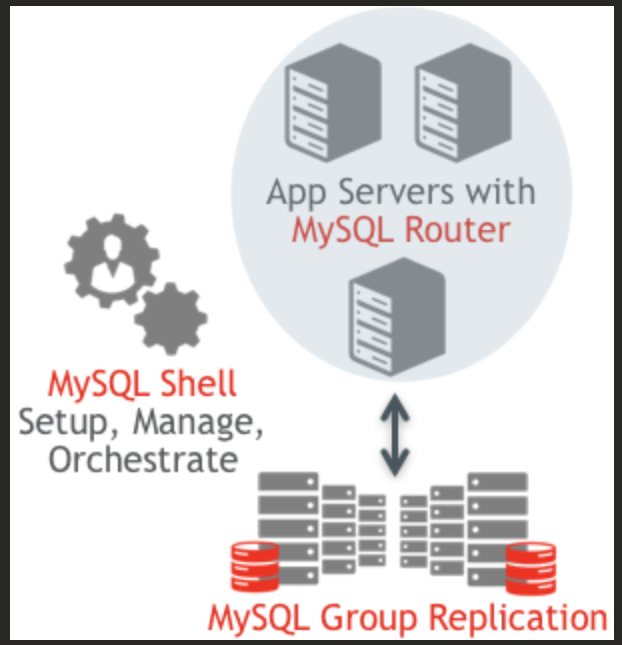


Node Types:

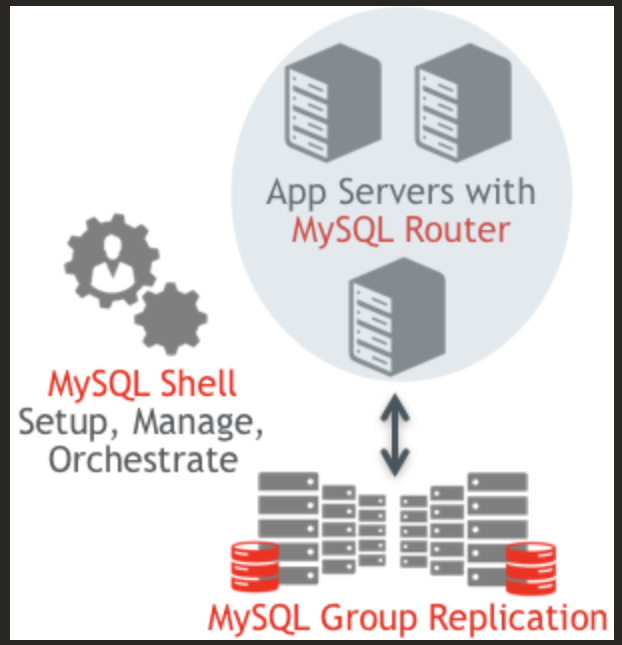
- **R**: Traffic proxy: MySQL Router
- **M**: mysqld nodes participating in MySQL Group Replication

MySQL Router

MySQL Router



MySQL Router



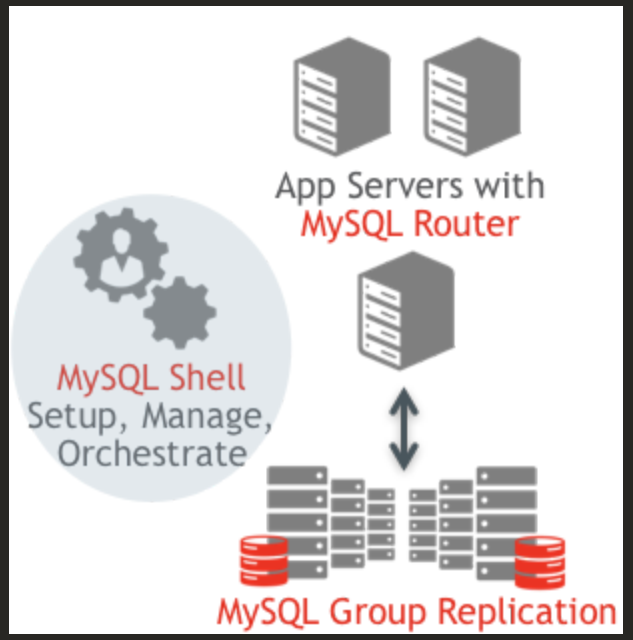
Transparent Access to Database Arch.

"provide transparent routing between your application and back-end MySQL Servers"

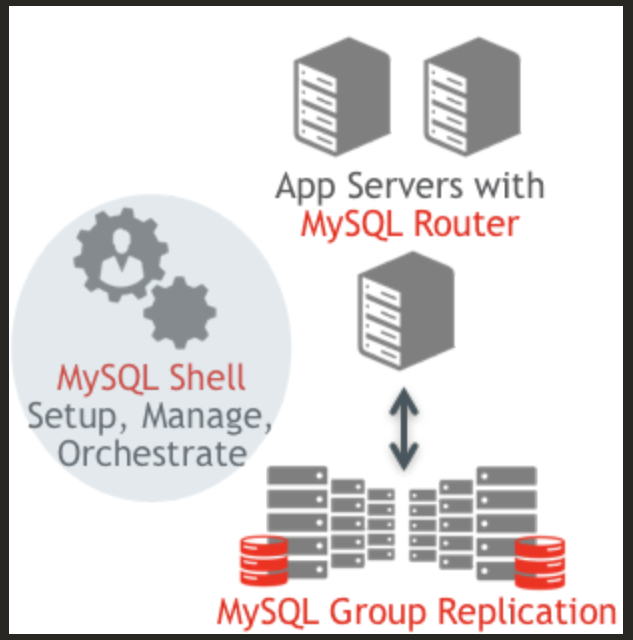
- Transparent client connection routing
  - Load balancing
  - Application connection failover
  - Little to no configuration needed
- Stateless design offers easy HA client routing
  - Router as part of the application stack
- Native support for InnoDB clusters
  - Understands Group Replication topology
- Currently TCP Port each for `PRIMARY` and `NON-PRIMARY` traffic

MySQL Shell

MySQL Shell


App Servers with
MySQL Router
MySQL Shell
Setup, Manage,
Orchestrate
MySQL Group Replication

MySQL Shell



Database Administration Interface

"MySQL Shell provides the developer and DBA with a single intuitive, flexible, and powerful interface for all MySQL related tasks!"

- Multi-Language: JavaScript, Python, and SQL
- Naturally scriptable
- Supports Document and Relational models
- Exposes full Development and Admin API
- Classic MySQL protocol and X protocol

MySQL Shell - Easy to Use

MySQL Shell - Easy to Use

Create a Cluster:

```
mysql-js> \c clusteradmin@mysql1
mysql-js> cluster = dba.createCluster('important_cluster')
```

MySQL Shell - Easy to Use

Create a Cluster:

```
mysql-js> \c clusteradmin@mysql1
mysql-js> cluster = dba.createCluster('important_cluster')
```

Configure server to add later:

```
mysql-js> dba.configureInstance('clusteradmin@mysql2')
```

MySQL Shell - Easy to Use

Create a Cluster:

```
mysql-js> \c clusteradmin@mysql1
mysql-js> cluster = dba.createCluster('important_cluster')
```

Configure server to add later:

```
mysql-js> dba.configureInstance('clusteradmin@mysql2')
```

Add server to the Cluster:

```
mysql-js> cluster.addInstance('clusteradmin@mysql2')
```

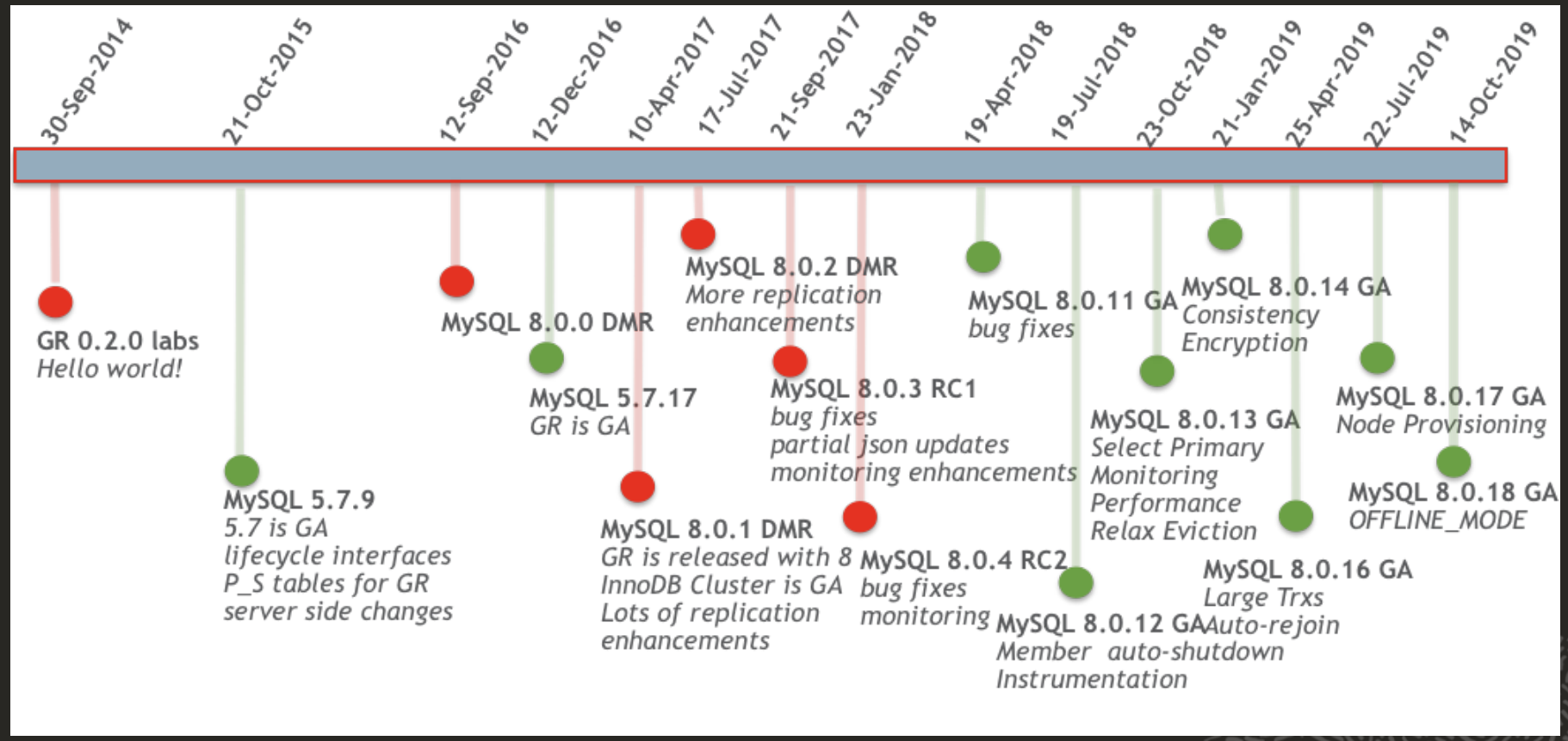
MySQL Shell - Easy to Use

Check the Cluster status:

```
mysql-js> cluster.status()
{
    "clusterName": "important_cluster",
    "defaultReplicaSet": {
        "name": "default",
        "primary": "mysql3:3306",
        "ssl": "REQUIRED",
        "status": "OK",
        "statusText": "Cluster is ONLINE and can tolerate up to ONE failure.",
        "topology": {
            "mysql2:3306": {
                "address": "mysql2:3306",
                "mode": "R/O",
                "readReplicas": {},
                "role": "HA",
                "status": "ONLINE"
            },
            "mysql3:3306": {
                "address": "mysql3:3306",
                "mode": "R/W",
                "readReplicas": {},
                "role": "HA",
                "status": "ONLINE"
...
```

MySQL InnoDB Cluster 8.0

MySQL InnoDB Cluster - Evolving



Reasons to use MySQL InnoDB Cluster 8.0

- MySQL Shell
- MySQL Router
- MySQL Server
- Diagnostics
- Network
- Network Partition Handling
- Dynamic Changes
- Configurable Consistency Levels
- Security
- Performance & Efficiency
- Node Provisioning

MySQL Shell 8.0.4 - Upgrade Checker

```
mysql-js> util.checkForServerUpgrade("root@localhost:3306")
The MySQL server at localhost:3306 will now be checked for compatibility issues for upgrade to MySQL 8.0..
MySQL version: 5.7.19 - MySQL Community Server (GPL)

1) Usage of db objects with names conflicting with reserved keywords in 8.0  -- No issues found
2) Usage of utf8mb3 charset
Warning: The following objects use the utf8mb3 character set. It is recommended to convert them to use utf8mb4 instead,
         for improved Unicode support.

simple_schema.city.name - column's default character set: utf8
simple_schema.city.country_code - column's default character set: utf8
3) Usage of use ZEROFILL/display length type attributes
Notice: The following table columns specify a ZEROFILL/display length attributes. Please be aware that they will be
        ignored in MySQL 8.0
test.big_table.ORDINAL_POSITION - bigint(21) unsigned
4) Issues reported by 'check table x for upgrade' command -- No issues found
5) Table names in the mysql schema conflicting with new tables in 8.0 -- No issues found
6) Usage of old temporal type -- No issues found
7) Foreign key constraint names longer than 64 characters -- No issues found

No fatal errors were found that would prevent a MySQL 8 upgrade, but some potential issues were detected.
Please ensure that the reported issues are not significant before upgrading.
```

https://mysqlserverteam.com/mysql-shell-8-0-4-introducing-upgrade-checker-utility/

MySQL Shell 8.0

- MySQL Shell 8.0.12: storing MySQL passwords securely

  https://mysqlserverteam.com/mysql-shell-8-0-12-storing-mysql-passwords-securely/

- Added new Shell options to simplify connections to InnoDB clusters

```
$ mysqlsh --redirect-primary
          --cluster
```

- MySQL Shell 8.0.13: Unix Shell command line syntax

```
$ mysqlsh myAdmin@localhost:3306 -- dba create-cluster testCluster --member-weight=65 \
      --exit-state-action=READ_ONLY
```

MySQL Shell 8.0

- MySQL Shell 8.0.12: storing MySQL passwords securely

  https://mysqlserverteam.com/mysql-shell-8-0-12-storing-mysql-passwords-securely/

- Added new Shell options to simplify connections to InnoDB clusters

```
$ mysqlsh --redirect-primary
          --cluster
```

- MySQL Shell 8.0.13: Unix Shell command line syntax

```
$ mysqlsh myAdmin@localhost:3306 -- dba create-cluster testCluster --member-weight=65 \
      --exit-state-action=READ_ONLY
```

Use MySQL Shell 8.0 even with 5.7 Group Replication!

Reasons to use MySQL InnoDB Cluster 8.0

- MySQL Shell
- MySQL Router
- MySQL Server
- Diagnostics
- Network
- Network Partition Handling
- Dynamic Changes
- Configurable Consistency Levels
- Security
- Performance & Efficiency
- Node Provisioning

MySQL Router 8.0

- MySQL Router 8.0.14: Dynamic Config

```
{
"metadata-cache": {
    "group-replication-id": "4b9e817a-0254-11e9-9cc0-080027bb5030",
    "cluster-metadata-servers": [
        "mysql://localhost:3310",
        "mysql://localhost:3320",
        "mysql://localhost:3330"
    ]
},
"version": "1.0.0"
}
```

https://dev.mysql.com/doc/mysql-router/8.0/en/mysql-router-conf-options.html#option_mysqlrouter_dynamic_config

- MySQL Router 8.0.17: Invalidate Metadata Cache based on Group Replication Notification
  X Protocol notifies Routers of Group Replication status/membership change.
  Best Practice with lots of Routers: `$ mysqlrouter --bootstrap --conf-use-gr-notifications`

MySQL Router 8.0 - REST API Monitoring

- 8.0.17: REST API Monitoring

```
curl -s http://host:8080/api/20190715/metadata | jq
{
  "items": [
    {
      "name": "myCluster"
    }
  ]
}
```

```
$ curl -s \
  http://host:8080/api/20190715/metadata/myCluster/config \
  | jq
{
  "clusterName": "myCluster",
  "timeRefreshInMs": 500,
  "groupReplicationId": "b2025e72-9e5e-11e9-95c9-08002718d305",
  "nodes": [
    {
      "hostname": "mysql1",
      "port": 3306
    },
    {
      "hostname": "mysql2",
      "port": 3306
    }
  ]
}
```

Use MySQL Router 8.0 even with 5.7 Group Replication!

Reasons to use MySQL InnoDB Cluster 8.0

- MySQL Shell
- MySQL Router
- MySQL Server
- Diagnostics
- Network
- Network Partition Handling
- Dynamic Changes
- Configurable Consistency Levels
- Security
- Performance & Efficiency
- Node Provisioning

MySQL Server 8.0 - Remote configureInstance()

```
dba.configureInstance("root@ic-1:3306")
Please provide the password for 'root@ic-1:3306': ***
Configuring MySQL instance at ic-1:3306 for use in an InnoDB cluster...

Some configuration options need to be fixed:
+--------------------------+---------------+----------------+-----------------------------------------------------+
| Variable                 | Current Value | Required Value | Note                                                |
+--------------------------+---------------+----------------+-----------------------------------------------------+
| binlog_checksum          | CRC32         | NONE           | Update the server variable                          |
| enforce_gtid_consistency | OFF           | ON             | Update read-only variable and restart the server    |
| gtid_mode                | OFF           | ON             | Update read-only variable and restart the server    |
+--------------------------+---------------+----------------+-----------------------------------------------------+

Do you want to perform the required configuration changes? [y/n]: y
Do you want to restart the instance after configuring it? [y/n]: y
Configuring instance...
The instance 'ic-1:3306' was configured for cluster usage.
Restarting MySQL...
MySQL server at ic-1:3306 was restarted.
```

```
SET PERSIST gtid_mode = ON;
RESTART
```

https://dev.mysql.com/doc/refman/8.0/en/persisted-system-variables.html

Reasons to use MySQL InnoDB Cluster 8.0

- MySQL Shell
- MySQL Router
- MySQL Server
- Diagnostics
- Network
- Network Partition Handling
- Dynamic Changes
- Configurable Consistency Levels
- Security
- Performance & Efficiency
- Node Provisioning

Diagnostics - Shell 8.0

- Shell 8.0.14: Extended cluster information

  https://mysqlserverteam.com/mysql-innodb-cluster-extended-cluster-information-and-rescan/

- Shell 8.0.14: Redesigned .rescan()

  https://mysqlserverteam.com/mysql-innodb-cluster-extended-cluster-information-and-rescan/

Diagnostics - Replication 8.0

- 8.0: Dynamic tracing and debugging log for Group Replication

```
group_replication_communication_debug_options="GCS_DEBUG_ALL,XCOM_DEBUG_TRACE"
```

- 8.0: Extending Group Replication performance_schema tables

  https://mysqlhighavailability.com/group-replication-extending-group-replication-performance_schema-tables/

- 8.0: Performance Schema Instrumentation for Group Replication

  https://mysqlhighavailability.com/more-p_s-instrumentation-for-group-replication/

- 8.0.1: New monitoring replication features

  https://mysqlhighavailability.com/new-monitoring-replication-features-and-more/

- 8.0.12: XCOM Cache instrumentation

  https://mysqlhighavailability.com/extending-replication-instrumentation-account-for-memory-used-in-xcom/

- 8.0.13: Replication Instrumentation: TRX Retries

  https://mysqlhighavailability.com/extending-replication-instrumentation-an-insight-on-transaction-retries/

Reasons to use MySQL InnoDB Cluster 8.0

- MySQL Shell
- MySQL Router
- MySQL Server
- Diagnostics
- Network
- Network Partition Handling
- Dynamic Changes
- Configurable Consistency Levels
- Security
- Performance & Efficiency
- Node Provisioning

Network

- 8.0.14: IPv6 Support
- 8.0.16: Enhanced Support for Large Transactions

```
group_replication_communication_max_message_size=10MB
```

https://mysqlhighavailability.com/enhanced-support-for-large-transactions-in-group-replication/

https://dev.mysql.com/doc/refman/8.0/en/group-replication-performance-message-fragmentation.html

- 8.0.2: Fine tuning options for the flow control mechanism

```
group_replication_flow_control_period
group_replication_flow_control_release_percent
```

Reasons to use MySQL InnoDB Cluster 8.0

- MySQL Shell
- MySQL Router
- MySQL Server
- Diagnostics
- Network
- Network Partition Handling
- Dynamic Changes
- Configurable Consistency Levels
- Security
- Performance & Efficiency
- Node Provisioning

Network Partition Handling

- 8.0.2: Configurable Member Weight (backported to 5.7.20)

```
group_replication_member_weight
```

- 8.0.12: Preventing Stale Reads

```
group_replication_exit_state_action=ABORT_SERVER/READ_ONLY/OFFLINE_MODE
```

https://mysqlhighavailability.com/fail-early-fail-fast-preventing-stale-reads-in-group-replication/

https://dev.mysql.com/doc/refman/8.0/en/group-replication-responses-failure.html

  - 8.0.18: `OFFLINE_MODE`

    https://mysqlhighavailability.com/automatic-member-fencing-with-offline_mode-in-group-replication/

    Best Practice: Leave default, let Router handle it

- 8.0.2: Configurable Majority Timeout

```
group_replication_unreachable_majority_timeout
```

Best Practice: Leave default

Network Partition Handling

- 8.0.16: Auto-rejoin: Automatically join back group after leaving/being expelled

```
group_replication_autorejoin_tries
```

https://mysqlhighavailability.com/no-ping-will-tear-us-apart-enabling-member-auto-rejoin-in-group-replication/

https://dev.mysql.com/doc/refman/8.0/en/group-replication-responses-failure.html

Best Practice: set `group_replication_autorejoin_tries=3`

- 8.0.13: Configurable Member Expel Timeout

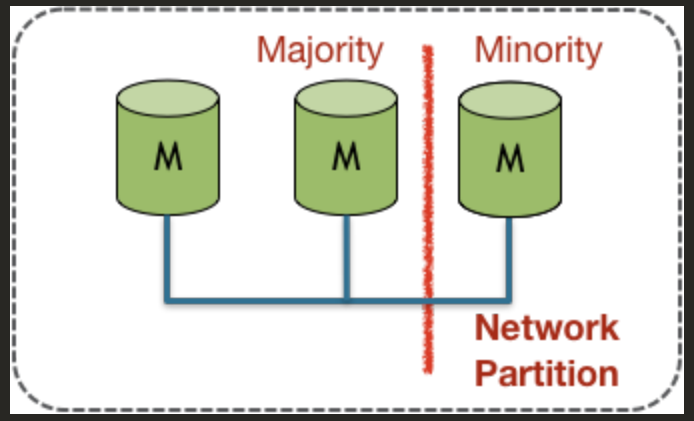
```
group_replication_member_expel_timeout
```

https://mysqlserverteam.com/mysql-innodb-cluster-controlling-data-consistency-and-expel-timeouts/

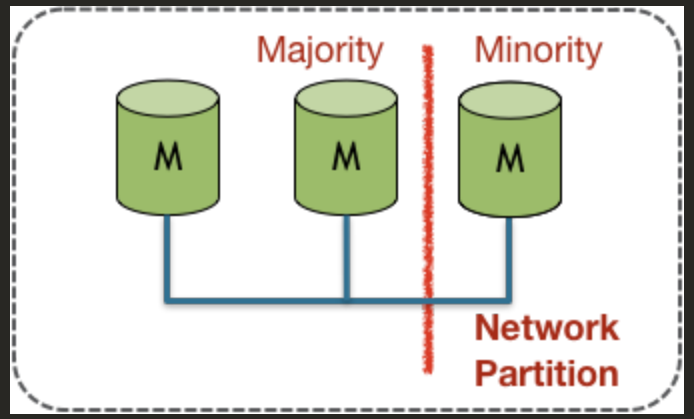
https://mysqlhighavailability.com/group-replication-coping-with-unreliable-failure-detection/

Best Practice: set `group_replication_member_expel_timeout=5`,
and with less reliable networks (public cloud), consider setting higher to `30-60`, but not too high!

MySQL Group Replication - Network Partitions

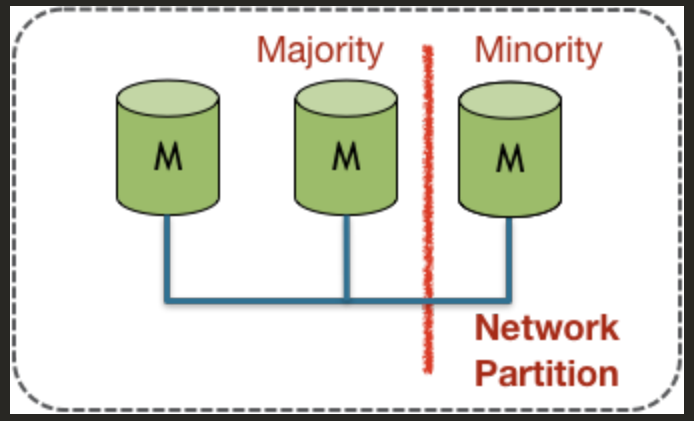


MySQL Group Replication - Network Partitions



- Quorum based Network Partition Handling
  - Prevents Split-Brain
  - Best Practice:
    Ensure uneven amount of nodes
- view of <=50% of total members: *Minority*
- view of >50% of total members: *Majority*
  - will accept reads/writes

MySQL Group Replication - Network Partitions



- Quorum based Network Partition Handling
  - Prevents Split-Brain
  - Best Practice:
    Ensure uneven amount of nodes
- view of <=50% of total members: *Minority*
- view of >50% of total members: *Majority*
  - will accept reads/writes

- In case of event of loss of majority:
  - a minority partition can be forced `ONLINE`
- Nodes in minority:
  - MySQL Router: moves away traffic
  - Fencing: members become `READ_ONLY`,`ABORT_SERVER`,`OFFLINE_MODE`

MySQL Group Replication - Network Partitions

Majority

1. wait 5 seconds, then suspect minority is gone
2. expel minority after `group_replication_member_expel_timeout` seconds (default `0`, immediately) (best: `5` seconds or ~`30`s with less reliable networks)
   During: members cannot be added/removed, no new `PRIMARY` can be elected
3. Elect a new `PRIMARY` if needed (if original `PRIMARY` is not part of `MAJORITY`)

Minority

1. wait 5 seconds before suspecting itself to lost connection to the majority
2. MySQL Router immediately disconnects existing connections to minority members.
3. wait `group_replication_unreachable_majority_timeout` (default `0`, forever) (best: `0`, forever)
4. when timeout: `group_replication_autorejoin_tries` (default `0` times) (best: 3 times)
   - retries every 5 minutes
5. After `group_replication_exit_state_action` (best: default `READ_ONLY`)
   - `READ_ONLY`: allow stale reads to be served (latest default)
   - `ABORT_SERVER`: Shutdown MySQL
   - `OFFLINE_MODE`: disallow connections to MySQL (except admins)

Reasons to use MySQL InnoDB Cluster 8.0

- MySQL Shell
- MySQL Router
- MySQL Server
- Diagnostics
- Network
- Network Partition Handling
- Dynamic Changes
- Configurable Consistency Levels
- Security
- Performance & Efficiency
- Node Provisioning

Dynamic Changes

- Shell 8.0.14: Changing cluster options “live”

  https://mysqlserverteam.com/mysql-innodb-cluster-changing-cluster-options-live/

- 8.0.13: Changing (Multi-)Primary Dynamically

Group Replication

```
group_replication_set_as_primary()
group_replication_switch_to_single_primary_mode()
group_replication_switch_to_multi_primary_mode()
```

https://mysqlserverteam.com/mysql-innodb-cluster-changing-cluster-topology-modes-live/

https://mysqlhighavailability.com/on-demand-primary-election/

https://dev.mysql.com/doc/refman/8.0/en/group-replication-changing-group-mode.html

Shell

```
setPrimaryInstance()
switchToSinglePrimaryMode()
switchToMultiPrimaryMode()
```

Reasons to use MySQL InnoDB Cluster 8.0

- MySQL Shell
- MySQL Router
- MySQL Server
- Diagnostics
- Network
- Network Partition Handling
- Dynamic Changes
- Configurable Consistency Levels
- Security
- Performance & Efficiency
- Node Provisioning

Configurable Consistency Levels

- 8.0.14: Consistent Reads

```
group_replication_consistency
```

https://mysqlhighavailability.com/group-replication-consistency-levels/

https://mysqlhighavailability.com/group-replication-consistent-reads/

https://mysqlhighavailability.com/group-replication-preventing-stale-reads-on-primary-fail-over/

https://mysqlhighavailability.com/group-replication-consistent-reads-deep-dive/

https://mysqlserverteam.com/mysql-innodb-cluster-controlling-data-consistency-and-expel-timeouts/

MySQL Group Replication - Consistency

Default consistency guarantee is `EVENTUAL`

- Write some data on `nodeA`,
- Read same data immediately on `nodeB`,
- data might be there yet... or not yet.

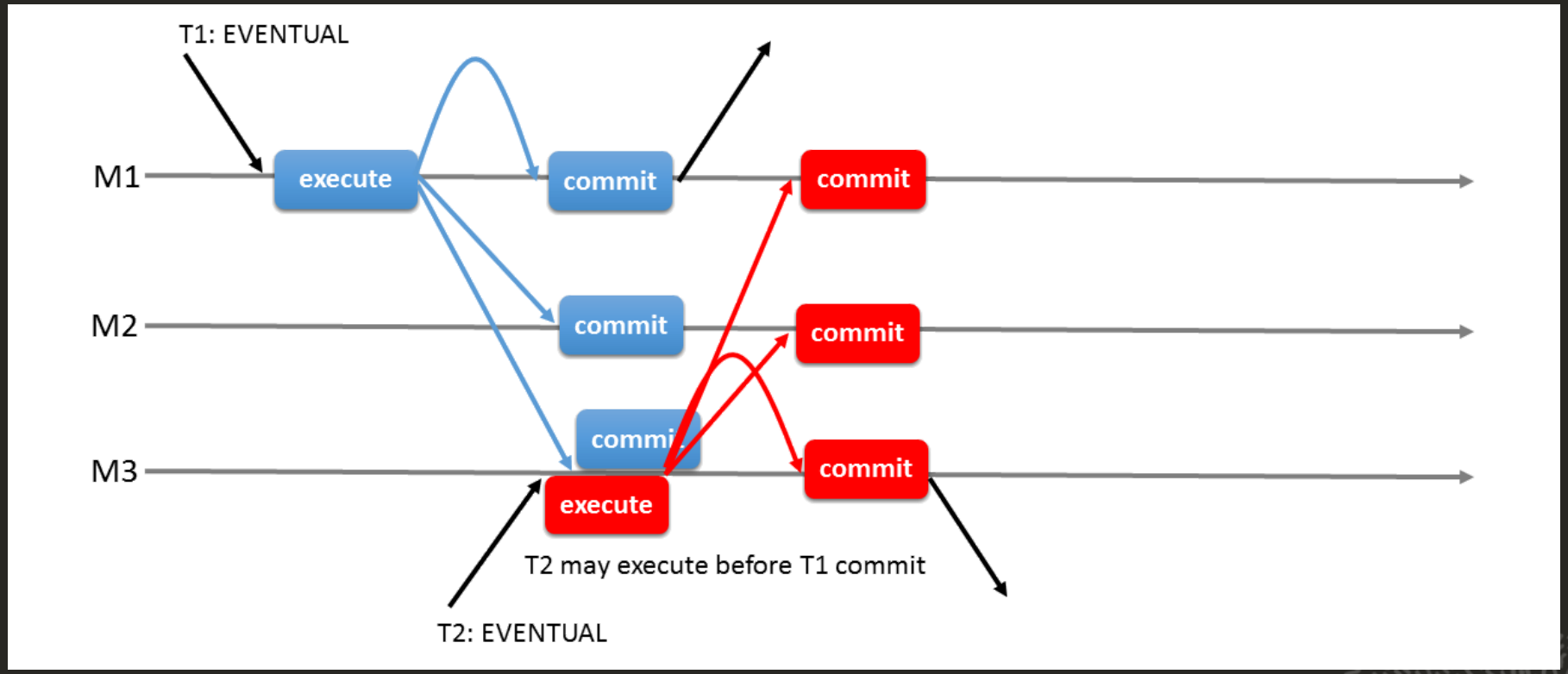
(Writes remain ordered and write consistency is never compromised)

Consistency level is configurable since MySQL 8.0.14 The scope can be `SESSION` or `GLOBAL`.

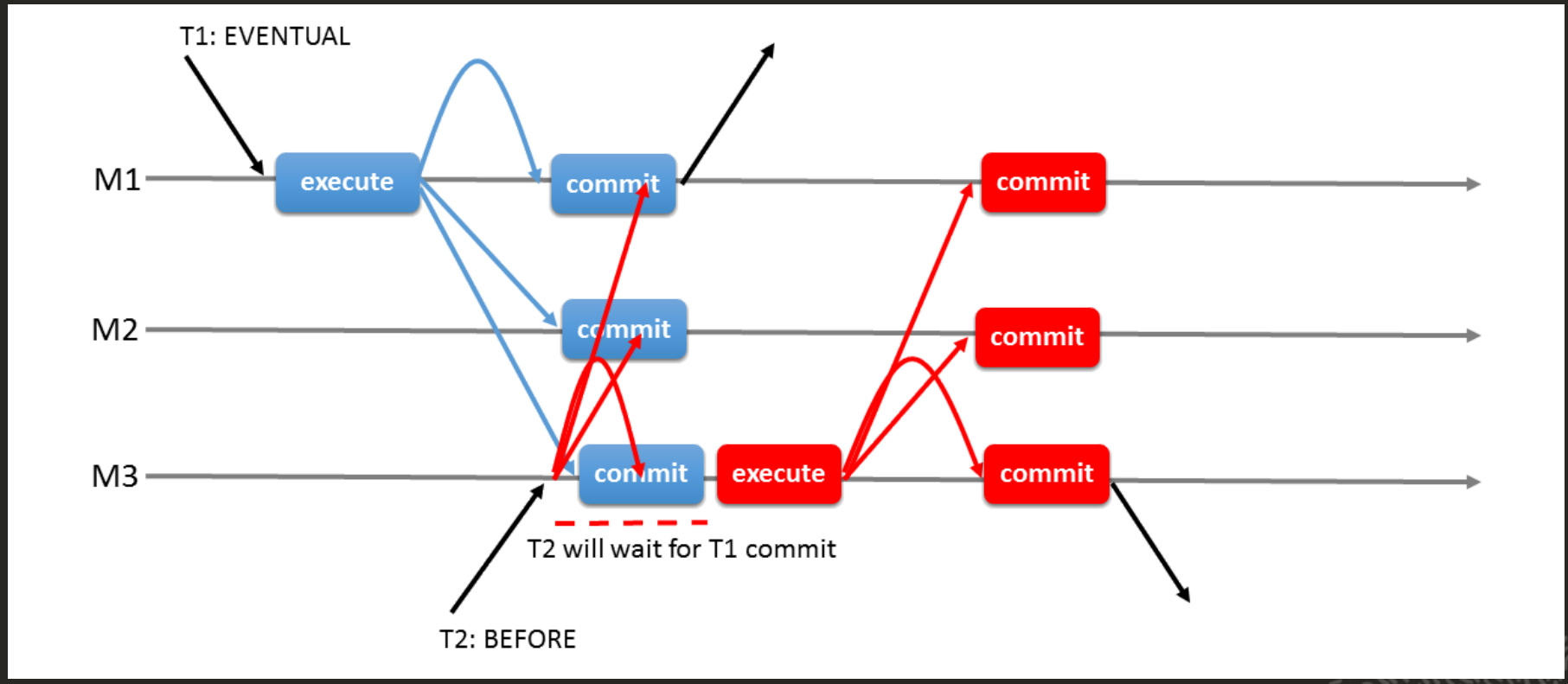
The possible values are:

- `EVENTUAL` (default)
- `BEFORE_ON_PRIMARY_FAILOVER`
- `BEFORE`
- `AFTER`
- `BEFORE_AND_AFTER`

MySQL Group Replication - Consistency - EVENTUAL

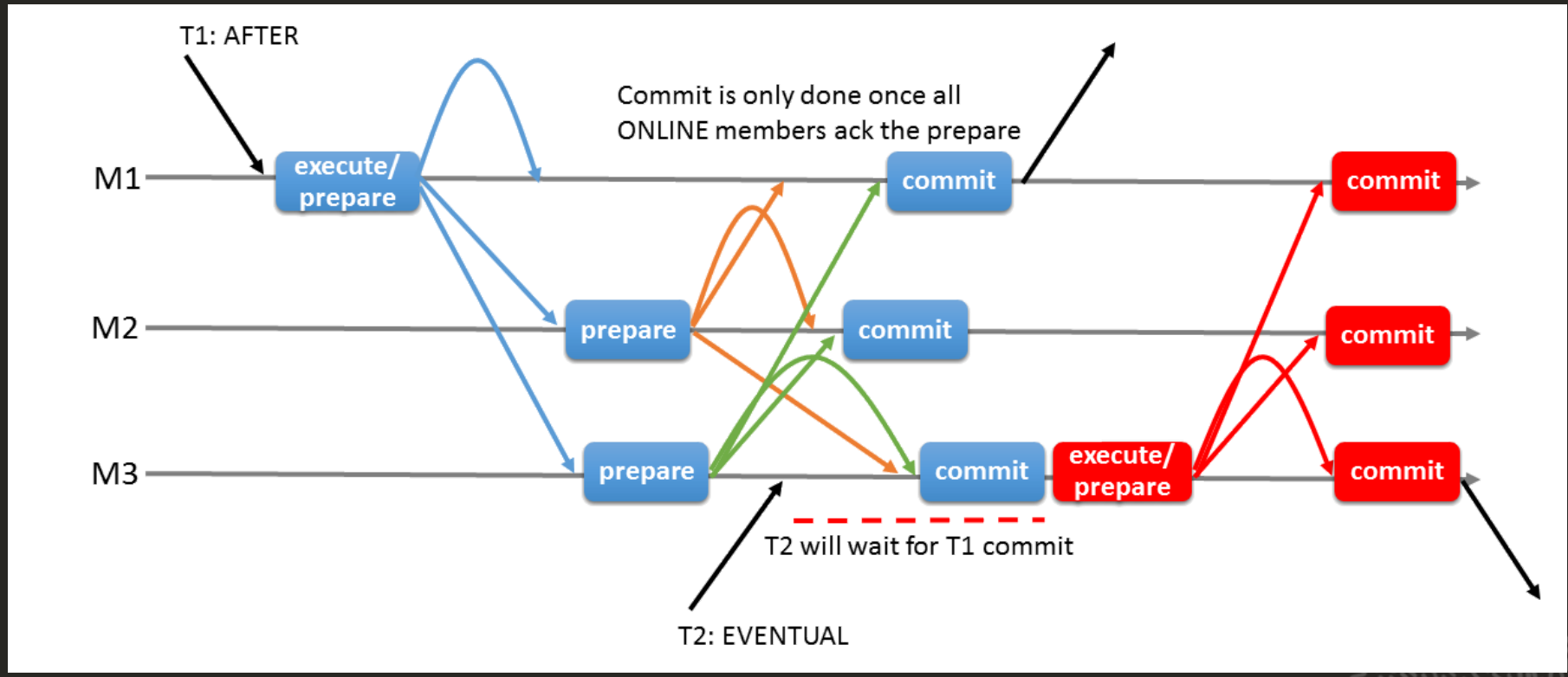


MySQL Group Replication - Consistency - BEFORE



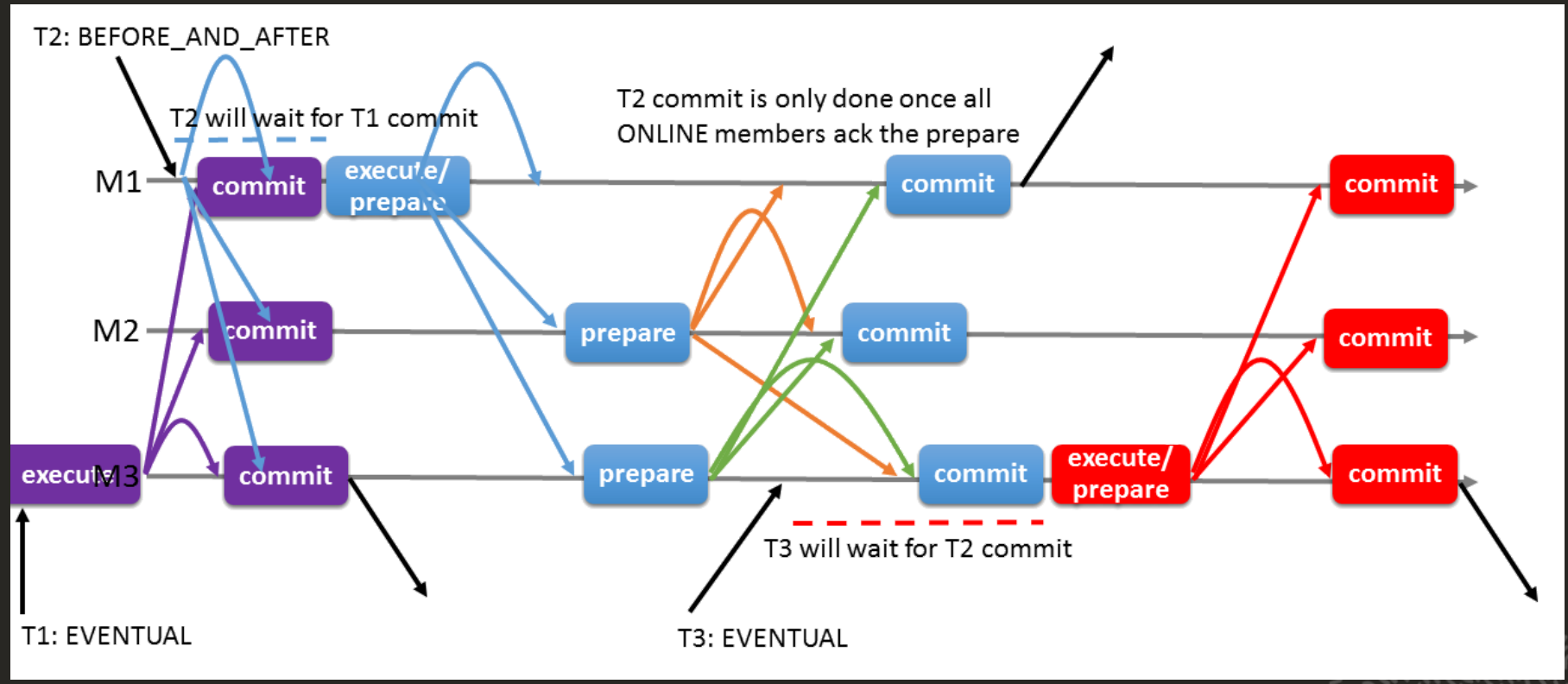
BEFORE READ or WRITE: wait for all trx to be applied
Impact on READ & WRITE: added response time: at least network latency

MySQL Group Replication - Consistency - AFTER



AFTER WRITE: wait for all trx to be applied
Impact on WRITE only: added response time: at least network latency

MySQL Group Replication - Consistency - BEFORE_AND_AFTER



BEFORE READ or WRITE: wait for all trx to be applied
AFTER WRITE: wait for all trx to be applied

Reasons to use MySQL InnoDB Cluster 8.0

- MySQL Shell
- MySQL Router
- MySQL Server
- Diagnostics
- Network
- Network Partition Handling
- Dynamic Changes
- Configurable Consistency Levels
- Security
- Performance & Efficiency
- Node Provisioning

Security

- 8.0.14: Binary Log Encryption https://mysqlhighavailability.com/binary-log-encryption-at-rest/

  https://mysqlhighavailability.com/how-to-manually-decrypt-an-encrypted-binary-log-file/
- 8.0.16: Binary Log Encryption Key Rotation

  https://mysqlhighavailability.com/rotating-binary-log-master-key-online/

Reasons to use MySQL InnoDB Cluster 8.0

- MySQL Shell
- MySQL Router
- MySQL Server
- Diagnostics
- Network
- Network Partition Handling
- Dynamic Changes
- Configurable Consistency Levels
- Security
- Performance & Efficiency
- Node Provisioning

Performance & Efficiency

- 8.0.1: replication performance improvements

  https://mysqlhighavailability.com/replication-performance-improvements-in-mysql-8-0-1/
- 8.0.1: Replication connection / applier enhancements

  https://mysqlhighavailability.com/no-new-options-no-new-commands-just-faster-at-full-load-where-it-counts/
- 8.0.3: Efficient JSON Replication in MySQL 8.0

  https://mysqlhighavailability.com/efficient-json-replication-in-mysql-8-0/
- 8.0: Performance Improvements

  https://mysqlhighavailability.com/performance-improvements-in-mysql-8-0-replication/
- 8.0.14: More efficient communication between GCS and XCOM

  https://mysqlhighavailability.com/group-replication-now-interacts-with-the-group-communication-engine-more-efficiently/
- 8.0.16: Tunable Paxos Message Cache for Group Replication.

  ```
  group_replication_message_cache_size=1G
  ```

  https://dev.mysql.com/doc/refman/8.0/en/group-replication-performance-xcom-cache.html

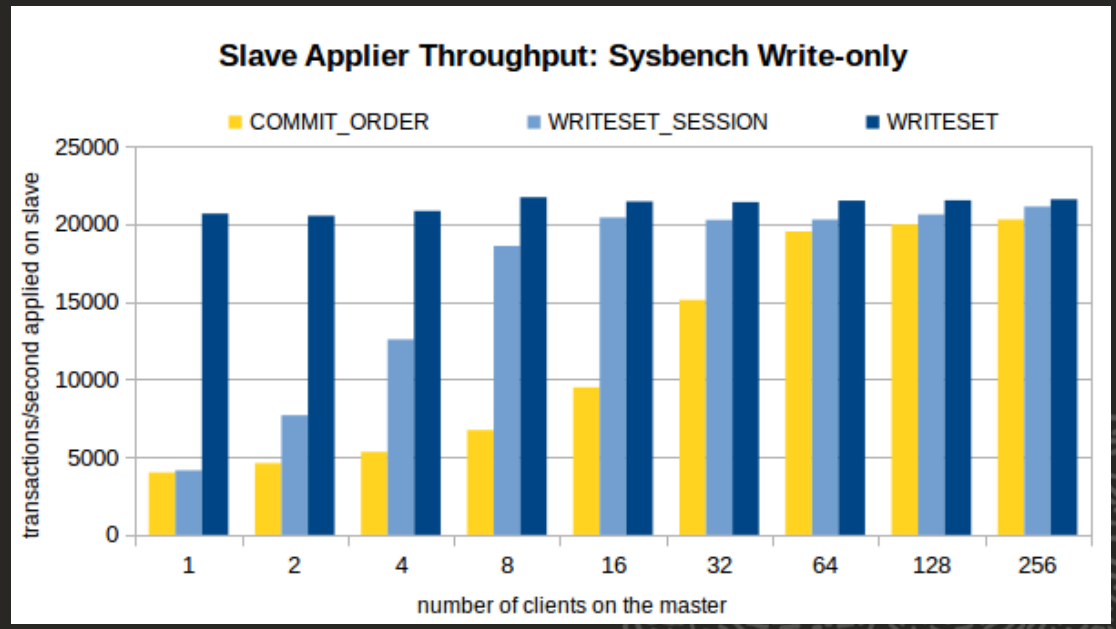
Performance & Efficiency - Writeset Based Replication

- 8.0.1: Writeset based Parallel Replication - Best Practice: Enable it!

  https://mysqlhighavailability.com/improving-the-parallel-applier-with-writeset-based-dependency-tracking/

```
binlog_transaction_dependency_tracking=WRITESET
```

  - used during distributed recovery & Group Replication
  - non conflicting transactions can be executed in parallel



Reasons to use MySQL InnoDB Cluster 8.0

- MySQL Shell
- MySQL Router
- MySQL Server
- Diagnostics
- Network
- Network Partition Handling
- Dynamic Changes
- Configurable Consistency Levels
- Security
- Performance & Efficiency
- Node Provisioning

8.0.17: Node Provisioning

- Provision members that have no data
- Takes Physical snapshot of data
- Using standard MySQL connection (3306)
- Built-in the MySQL Server
- Fully integrated in MySQL InnoDB Cluster

https://mysqlhighavailability.com/mysql-innodb-cluster-automatic-node-provisioning

https://mysqlhighavailability.com/a-breakthrough-in-usability-automatic-node-provisioning

8.0.17: Node Provisioning

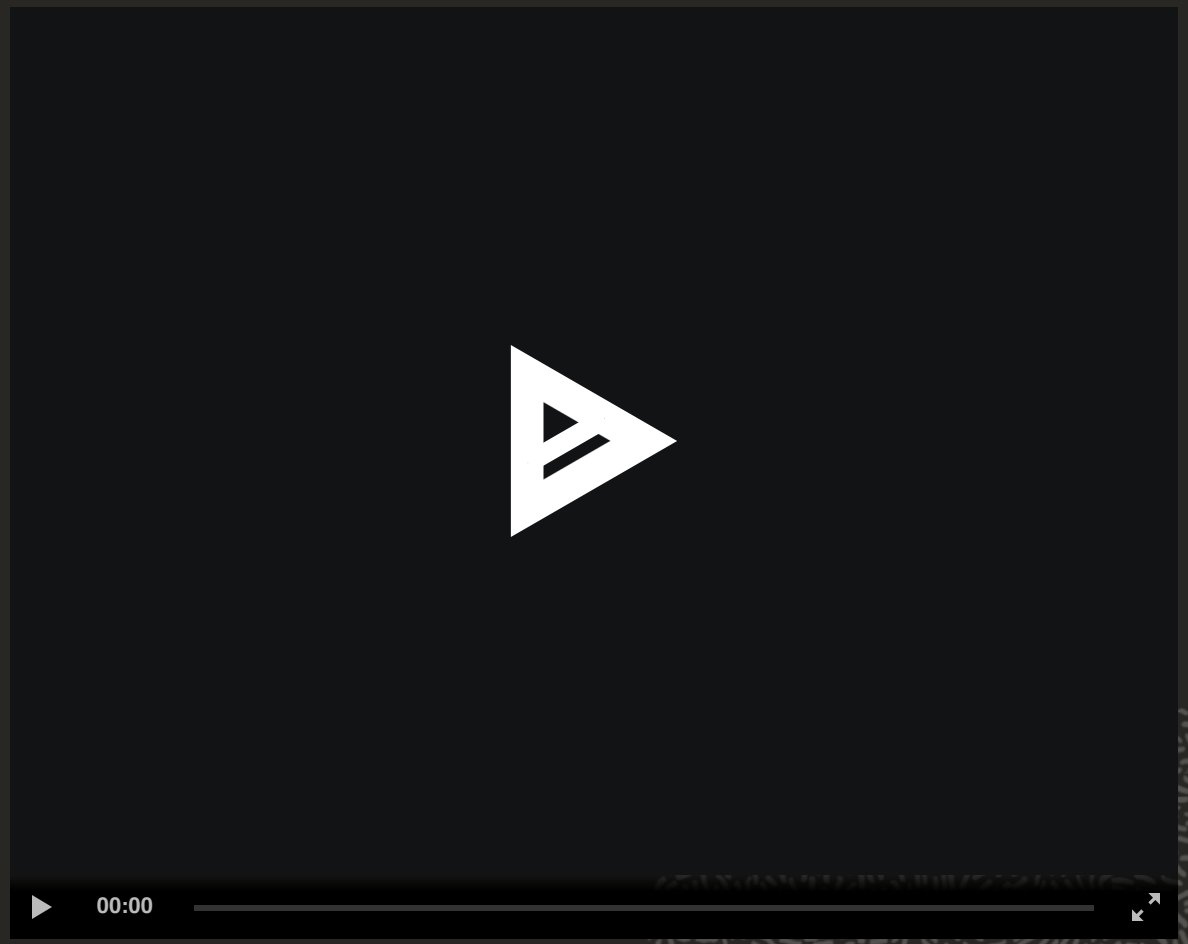
```
mysqlsh-js> cluster.addInstance("root@10.175.165.243:3320")
Please provide the password for 'root@10.175.165.243:3320': ******
Save password for 'root@10.175.165.243:3320'? [Y]es/[N]o/Ne[v]er (default No):
NOTE: A GTID set check of the MySQL instance at '10.175.165.243:3320' determined that
it is missing transactions that were purged from all cluster members.

Please select a recovery method [C]lone/[A]bort (default Abort): c
Validating instance at 10.175.165.243:3320...
NOTE: Instance detected as a sandbox.
Please note that sandbox instances are only suitable for deploying test clusters for use within the same host.

This instance reports its own address as 10.175.165.243:3320

Instance configuration is suitable.
A new instance will be added to the InnoDB cluster. Depending on the amount of
data on the cluster this might take from a few seconds to several hours.

Monitoring recovery process of the new cluster member. Press ^C to stop monitoring and let it continue in background.
Clone based state recovery is now in progress.

NOTE: A server restart is expected to happen as part of the clone process. If the
server does not support the RESTART command or does not come back after a
while, you may need to manually start it back.
```

8.0.17: Node Provisioning

```
 * Waiting for clone to finish...
NOTE: 10.175.165.243:3320 is being cloned from 10.175.165.243:3310
 ** Stage DROP DATA: Completed
 ** Clone Transfer
FILE COPY  ##################################################  100%  Completed
PAGE COPY  ##################################################  100%  Completed
REDO COPY  ##################################################  100%  Completed
 ** Stage RECOVERY: \
NOTE: 10.175.165.243:3320 is shutting down...

 * Waiting for server restart... ready
 * 10.175.165.243:3320 has restarted, waiting for clone to finish...
 * Clone process has finished: 3.66 GB transferred in 10 sec (366.14 MB/s)

State recovery already finished for '10.175.165.243:3320'

The instance '10.175.165.243:3320' was successfully added to the cluster.
```

8.0.17: Node Provisioning

Demo available on https://mysqlhighavailability.com/mysql-innodb-cluster-automatic-node-provisioning



Key Reference Blog Posts

MySQL InnoDB Cluster: What's New

mysqlserverteam.com/mysql-innodb-cluster-whats-new-in-the-8-0-ga-release/

mysqlserverteam.com/mysql-innodb-cluster-whats-new-in-the-8-0-13-ga-release/

mysqlserverteam.com/mysql-innodb-cluster-whats-new-in-the-8-0-14-ga-release/

mysqlserverteam.com/mysql-innodb-cluster-whats-new-in-the-8-0-16-release/

mysqlserverteam.com/mysql-innodb-cluster-whats-new-in-the-8-0-17-release/

mysqlserverteam.com/mysql-innodb-cluster-whats-new-in-shell-adminapi-8-0-18-release/

MySQL Replication: What's New

mysqlhighavailability.com/mysql-8-0-new-features-in-replication/

mysqlhighavailability.com/performance-improvements-in-mysql-8-0-replication/

mysqlhighavailability.com/replication-performance-improvements-in-mysql-8-0-1/

mysqlhighavailability.com/replication-features-in-mysql-8-0-1/

mysqlhighavailability.com/replication-features-in-mysql-8-0-2/

mysqlhighavailability.com/replication-features-in-mysql-8-0-3/

mysqlhighavailability.com/replication-features-in-mysql-8-0-4/

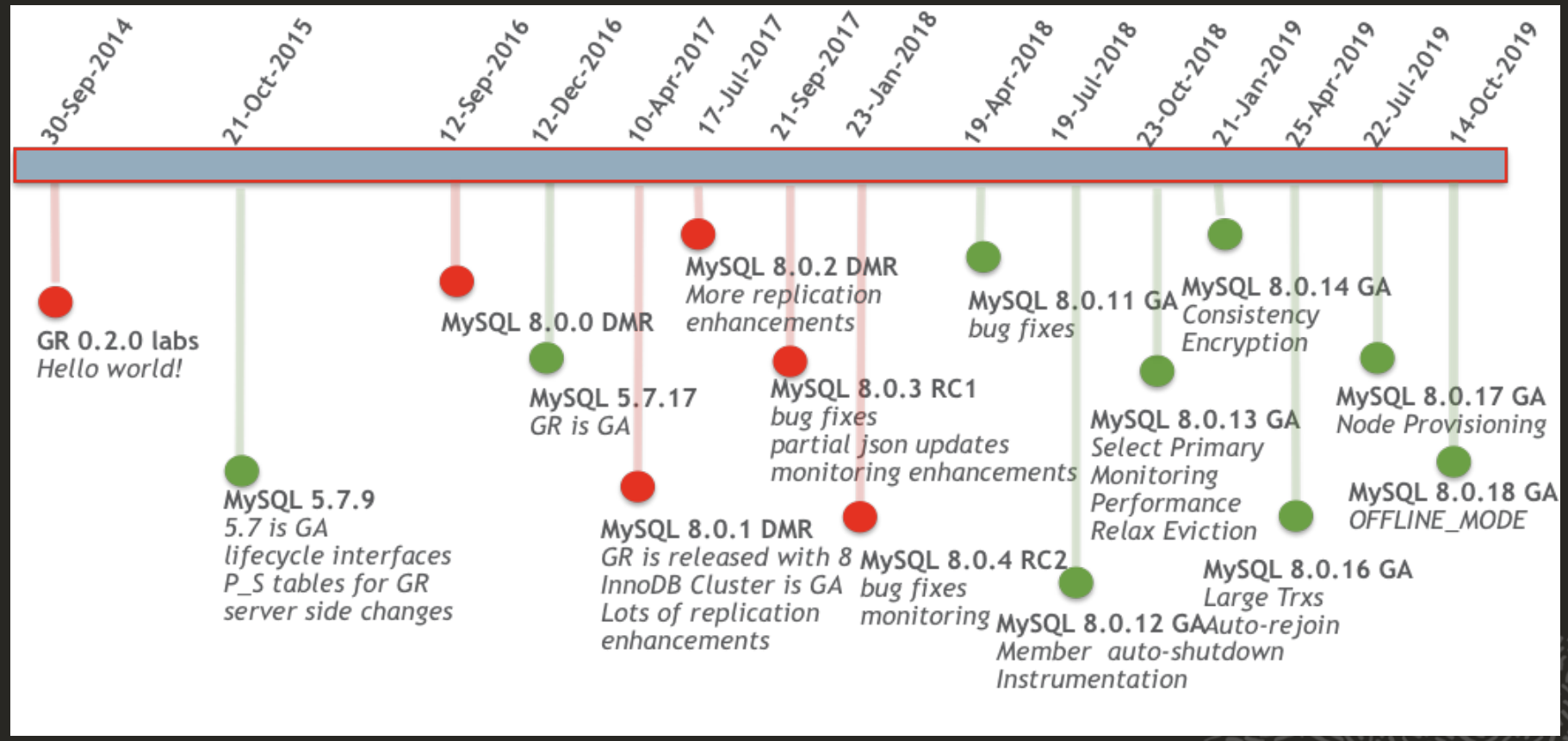
mysqlhighavailability.com/mysql-8-0-14-replication-enhancements/

mysqlhighavailability.com/mysql-8-0-16-replication-enhancements/

mysqlhighavailability.com/mysql-8-0-17-replication-enhancements/

mysqlhighavailability.com/mysql-8-0-18-replication-enhancements/

MySQL InnoDB Cluster - Evolving



Best Practice: Time to upgrade to 8.0!

Questions ?

https://dev.mysql.com/
http://www.mysql.com
https://mysqlhighavailability.com/
https://mysqlserverteam.com/