MySQL Replication – Advanced Features

In 20 minutes

Peter Zaitsev, CEO

FOSDEM, Brussels, Belgium
February 2nd, 2019

Question #1

Who in this room is using some kind of MySQL Replication ?

Question #2

Which Type of MySQL Replication do you use

- Classical MySQL Replication
- MySQL Group Replication
- Galera Replication (including Percona XtraDB Cluster)

Replication in MySQL

A brief Time Line

MySQL Replication

MySQL 3.23	MySQL 4.0	MySQL 5.1	MySQL 5.6	MySQL 5.7
• Initial Statement Replication Implemented	• Split IO Thread and SQL Thread	• Row and Mixed replication modes supported	• Per Database Parallel Replication • GTID Support	• General Parallel Replication • Multi-Source Replication • Group Replication / MySQL Innodb Cluster

Alternative Track

Galera Based Replication Technology

Similar to MySQL Group Replication but more mature

Available for MySQL, Percona, MariaDB

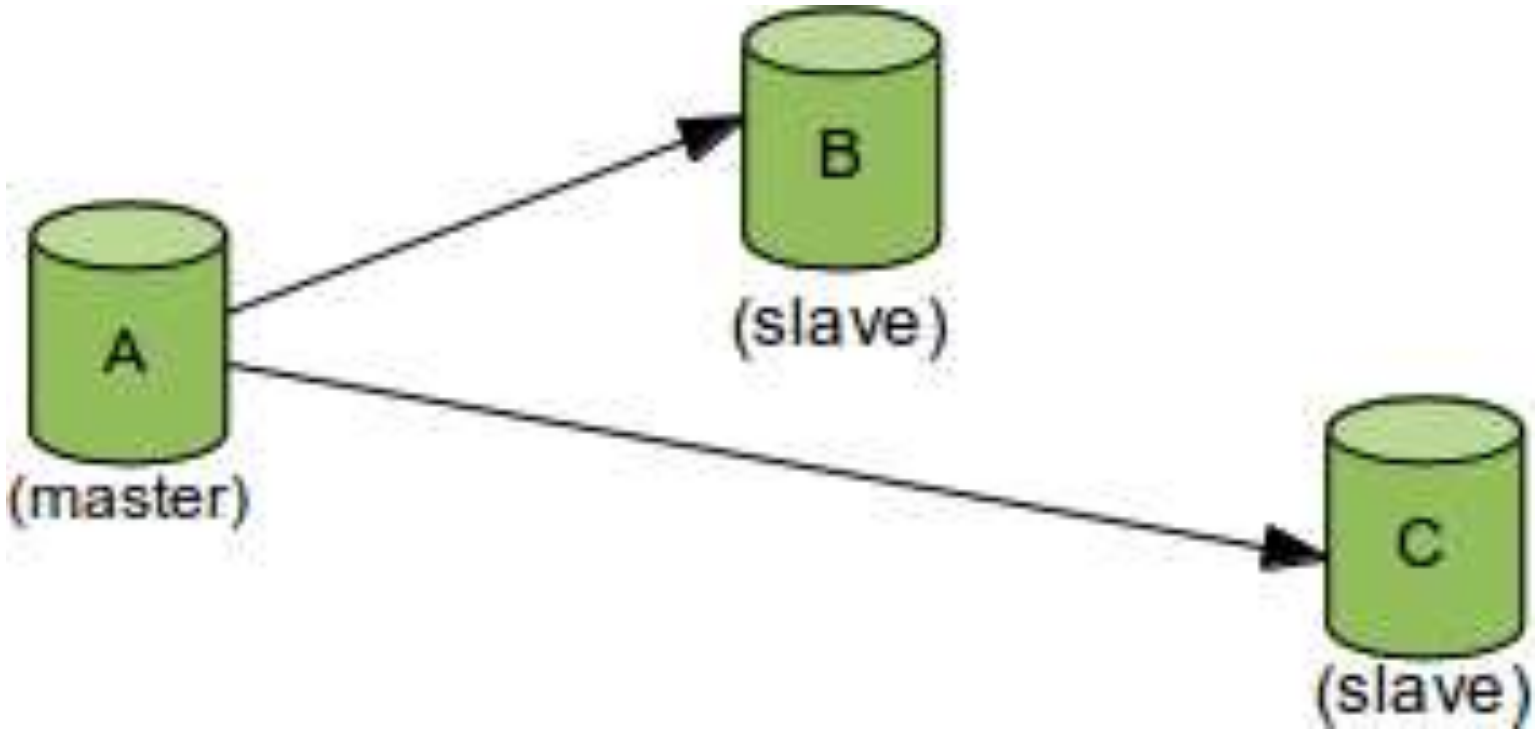
PERCONA XtraDB Cluster

Replication in MariaDB

- Is not 100% Same as in MySQL
- Different GTID Implementation
- Different parallel replication
- Not instrumented in the same way
- Number of features became available in MariaDB Earlier
- Not covering MariaDB in this Presentation

Advanced Replication

Basic Replication Topology



What is Basic Replication ?

- Statement Replication
- Replication of the Full Database
- Using Binary Log Position
- Single Tier Topology
- Single Thread
- Asynchronous
- Write to Single Master
- Keep Slave Meta-Data Data in Files
- Master-Slave-Replication
- Manual Failover & Traffic Management

Statement vs Row Replication

Configured on the Master as binlog_format

- STATEMENT
- ROW
- MIXED

How Rows are Logged ?

binlog_row_image

- Full
- Minimal
- Noblob

Missing Seeing Queries ?

Also Log Query For informational Purposes

binlog_rows_query_log_events

Off by Default

Replication of Full Database

Full Replication Most Simple for Troubleshooting and Recovery

Can replicate only portions of the database

Can add additional data to the slaves

Replication Filering

On the Master (writing binary log)

- Binlog_do_db
- Binlog_ignore_db

On the Slave

- Replicate_do_db
- Replicate_ignore_db
- Replicate_wild_do_table
- Replicate_wild_ignore_table

Position Identification

By Binary Log Position

- **File:** mysql-bin.000003
- **Position:** 73

By GTID

- 3E11FA47-71CA-11E1-9E33-C80AA9429562:23

What is in GTID ?

GTID = source_id:transaction_id

Source_id is Server_uuid of Originating Server

Always Preserved on the Slaves

Track which transactions executed

mysql.gtid_executed table (5.7) + binlog

GTID Benefits and Drawbacks

Automatic Position Discovery

Easy Slave Promotion

Discovery of Missed Transactions

Pain in the Ass with Manual Troubleshooting

Replication Topologies

- Single Tier Master-Slave
- Bi-Directional
- Tree
- Ring
- Directed Graph

Options For Complicated Topologies

Log_slave_updates

- Store copy of applied statements in local binlog
- Disabled by default until MySQL 8

Parallel Replication

Single Thread Replication is typical limiting factor

Parallel Replication for multiple Databases since 5.6

Parallel Replication for same table since 5.7

Managing Parallel Replication

slave-parallel-workers

slave-parallel-type=DATABASE|LOGICAL_CLOCK

slave_preserve_commit_order

Standard MySQL Replication - Asynchronous

Data Persisted on the Master

… Eventually Transferred to the Slave

After that Eventually Applied on the Slave

Replication Lag is Unbound

Data Loss can happen on Master Loss

Semi-Synchronous Replication in MySQL

Plugin Required

- Both on Master and Slave

Master

- INSTALL PLUGIN rpl_semi_sync_master SONAME 'semisync_master.so';

Slave

- INSTALL PLUGIN rpl_semi_sync_slave SONAME 'semisync_slave.so';

Semi-Sync Replication Options

Master: rpl_semi_sync_master_enabled

Master: rpl_semi_sync_master_timeout

Master: rpl_semi_sync_master_wait_point=AFTER_SYNC|AFTER_COMMIT

Master: *rpl_semi_sync_master_wait_for_slave_count*

Slave: rpl_semi_sync_slave_enabled

MySQL Replication Active Multi-Master

Recipe for problems with Async/SemiSync Replication

Design Application to Avoid Conflicts

MySQL Group Replication

PXC/Galera Based Solutions

Multi-Master with MySQL Classic Replication

auto_increment_offset

auto_increment_increment

slave_exec_mode=idempotent

Replication Position information

- Master_info_repository=FILE|TABLE
- Relay_log_info_repository=FILE|TABLE
- Sync_master_info
- Sync_relay_log_info
- Relay_log_recovery
- Less Problem with GTID Replication

Master Slave Replication ?

Most Commonly Used Variant

MySQL Group Replication is other Option

Percona XtraDB Cluster/Galera - Community Alternative

MySQL Group Replication Overview

Inspired by Galera Ideas (and Success)

Built on top of standard MySQL Replication

Available as Plugin for MySQL 5.7

Considered GA, Very Actively Developed

Difference from MySQL Replication

- No Master/Slave but Group Membership
- Transactions are committed when they are certified by majority of nodes (Paxos)
- Does not accept writes if there is no Quorum
- Flow Control to prevent unlimited replication lag
- Nodes encountering inconsistency leave the cluster
- Conflict Detection and Resolution (or avoidance)
- Simple FailOver

MySQL Group Replication - Writes

By Default configures itself as Single Primary

Can configure to allow writes to any node

MySQL Group Replication Limitations

No Automated node Provisioning

Manual Recovery of nodes with network failure

~~No way (yet) to ensure node does not read stale data~~

MySQL Innodb Cluster

Advanced Replication tools to Consider

Automating Failover	• Orchestrator
Read Write Splitting and Traffic Management	• ProxySQL
Sharding	• Vitess

Database Performance Matters