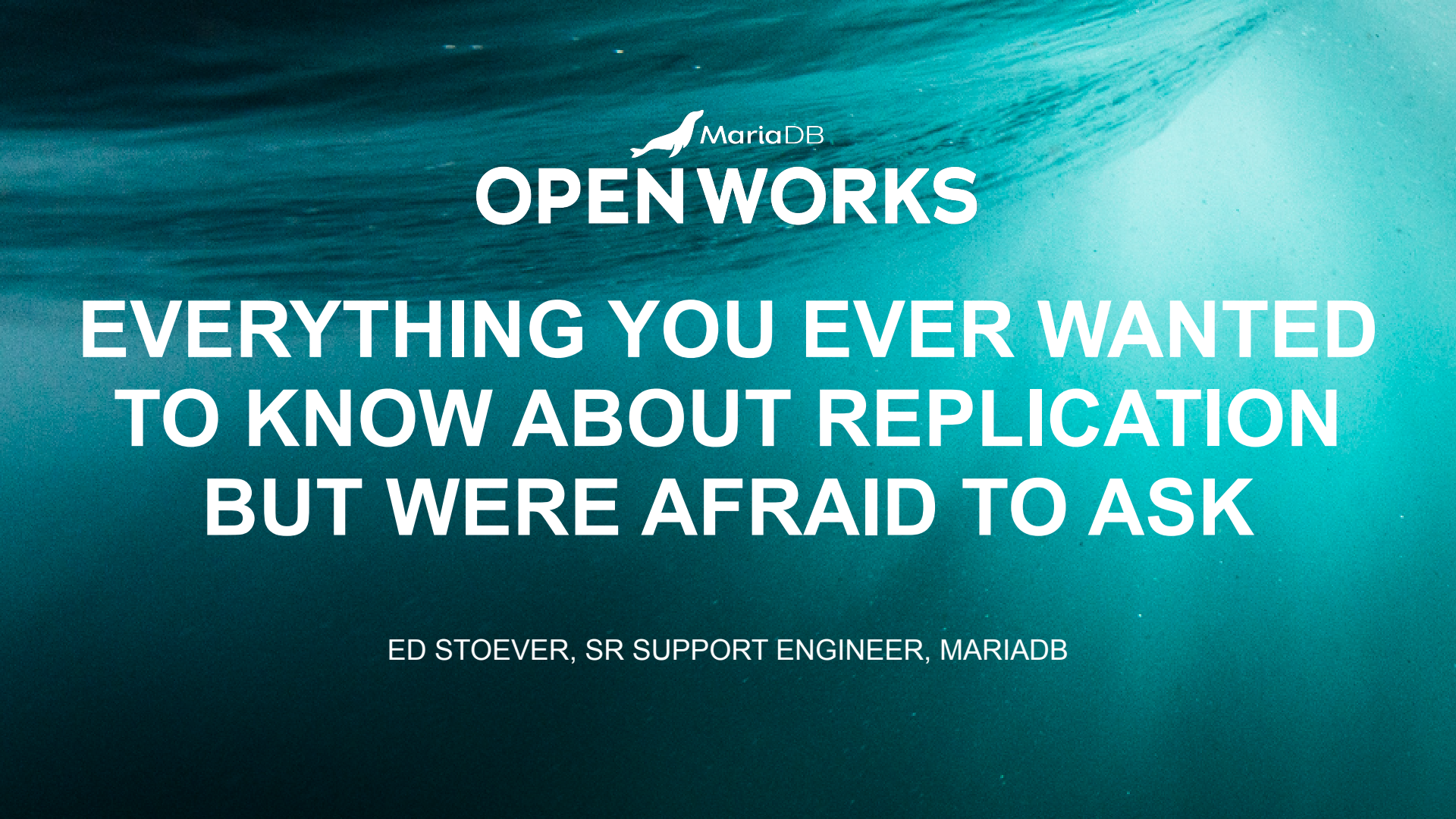

MariaDB
OPEN WORKS
BE UNSTOPPABLE



EVERYTHING YOU EVER WANTED TO KNOW ABOUT REPLICATION BUT WERE AFRAID TO ASK

ED STOEVER, SR SUPPORT ENGINEER, MARIADB

WHAT IS REPLICATION?

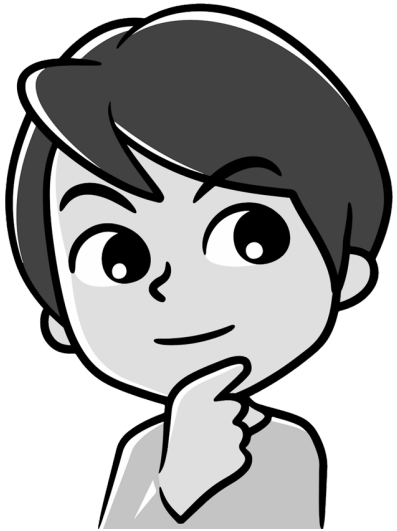
Primary/Master

Replica/Slave

MariaDB

WHY USE REPLICATION?

- Divide the work among multiple servers by running DML on the primary and queries on replicas
- Replicas are able to work without adversely affecting the primary
- Have backup server(s) ready at all times for High Availability
- Replication can be maintained across geographic regions
- You can replicate certain tables, certain schemas, or everything
- A variety of possible topologies
- Host machines do not have to be similar
- Replication can be configured with an intentional delay, a method for reviewing the primary as it was in the past: CHANGE MASTER TO master_delay=43200;



Asynchronous Replication

Replicas request events from the primary's binary log whenever the replicas are ready. The primary does not wait for a replica to confirm that an event has been received.

Semi-synchronous Replication

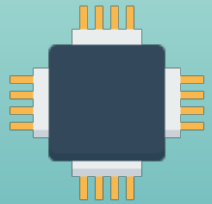
Semi-synchronous replication waits for just one replica to acknowledge that it has received and logged the events.
This feature can be enabled dynamically with:
set global rpl_semi_sync_master_enabled = ON;
set global rpl_semi_sync_slave_enabled = ON;

Fully Synchronous Replication

All replicas are required to respond that they have received the events. Galera cluster is an example of this.

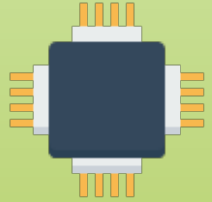
THREADS

Primary/Master



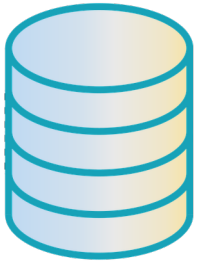
- Binary Log Dump Thread
- ACK Receiver Thread
  - (semi-sync only)

Replica/Slave

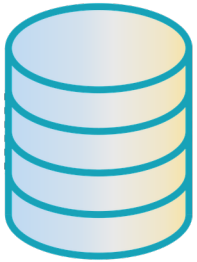


- Slave I/O Thread
- Slave SQL Thread
  - Can be multi-threaded
  - slave_parallel_threads=4
  - Worker threads

KEY CONCEPT #1 GTIDS



1-1-111

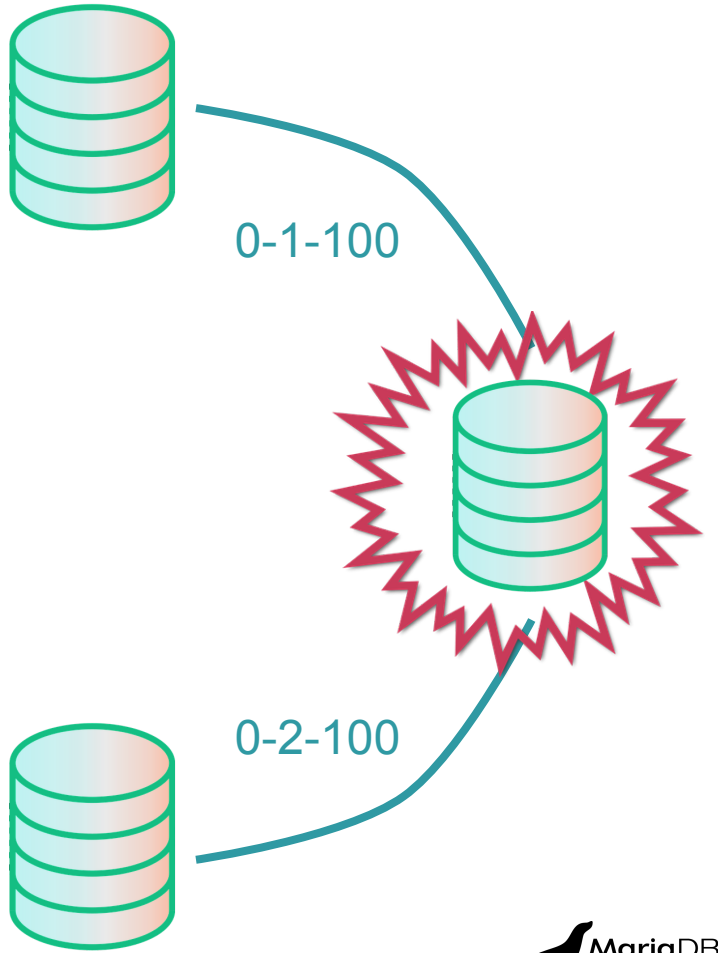


2-2-201

gtid_domain_id = 1, server_id = 1, gtid_seq_no = 111
gtid_domain_id = 2, server_id = 2, gtid_seq_no = 201

MariaDB

gtid_seq_no steps higher with each transaction and relies on gtid_domain_id to avoid conflicts. If separate primary servers cannot ensure an increasing and sequential gtid_seq_no, they should be configured to use distinct gtid_domain_id's.

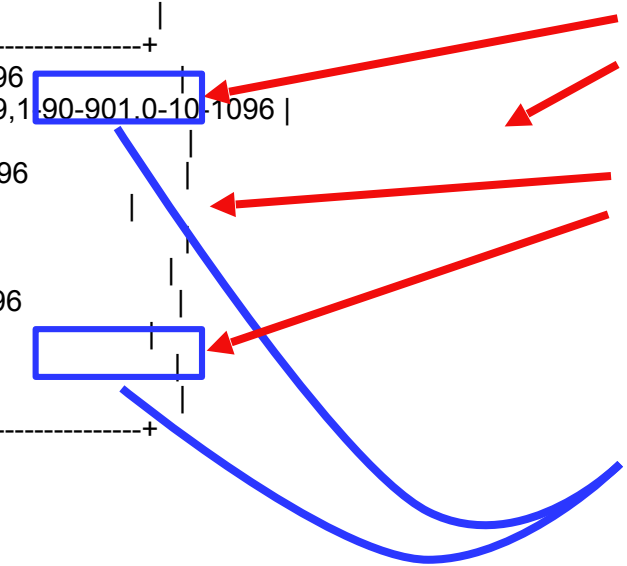


MariaDB

UNDERSTANDING GTID GLOBAL VARIABLES

We are looking at a replica. Primary server_id=10, replica server_id=20.

```
MariaDB [dranapp]> show global variables like '%gtid%';
+-----------------------+--------------------------+
| Variable_name         | Value                    |
+-----------------------+--------------------------+
| gtid_binlog_pos       | 0-10-1096                |
| gtid_binlog_state     | 0-20-159,1-90-901,0-10-1096 |
| gtid_cleanup_batch_size | 64                     |
| gtid_current_pos      | 0-10-1096                |
| gtid_domain_id        | 0                        |
| gtid_ignore_duplicates | OFF                     |
| gtid_pos_auto_engines |                          |
| gtid_slave_pos        | 0-10-1096                |
| gtid_strict_mode      | ON                       |
| wsrep_gtid_domain_id  | 0                        |
| wsrep_gtid_mode       | OFF                      |
+-----------------------+--------------------------+
11 rows in set (0.001 sec)
```



- The last event group written to the binary log
- Every combination of domain_id and server_id. Normally this internal state is not needed by users. Historic.
- The last transaction applied to the database
- The last transaction applied to the database by the server's replica threads (relay log).

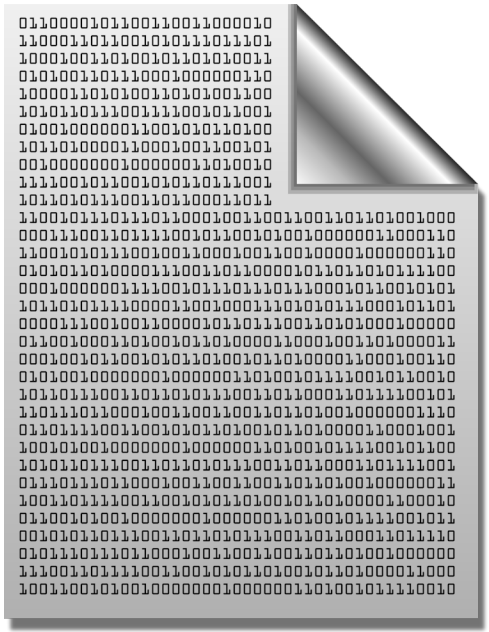
When gtid_binlog_pos and gtid_slave_pos are equal, it means log_slave_updates = 1

KEY CONCEPT #2 BINARY LOGS AND RELAY LOGS

log_slave_updates = [0|1]

MariaDB

BINARY LOG FORMAT



The binlog_format global variable will determine the format of both binary logs and relay logs.

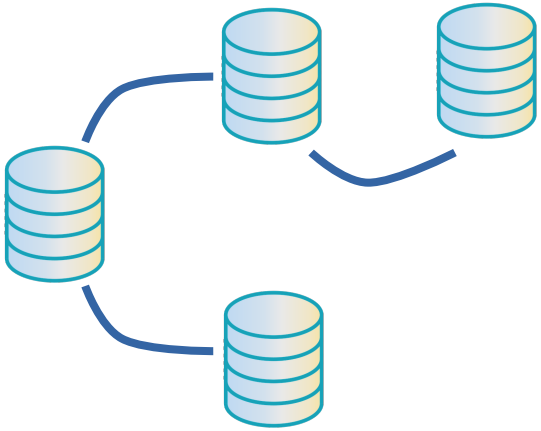
- binlog_format=STATEMENT
  - Not deterministic
- binlog_format=MIXED
- binlog_format=ROW

Not deterministic:
\INSERT INTO my_table (col1) VALUES (rand());

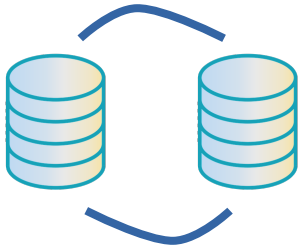
https://mariadb.com/kb/en/unsafe-statements-for-statement-based-replication/

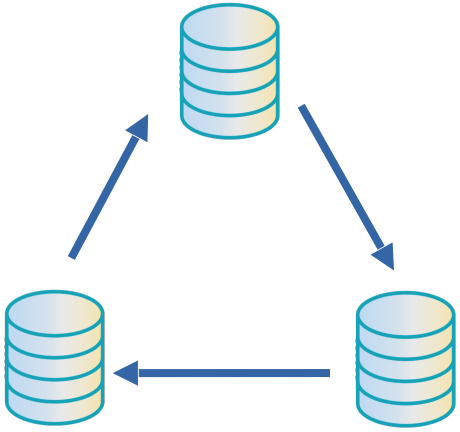
TOPOLOGIES

Linear Topologies



Circular Topologies

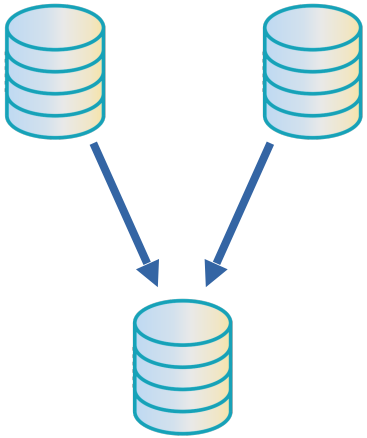




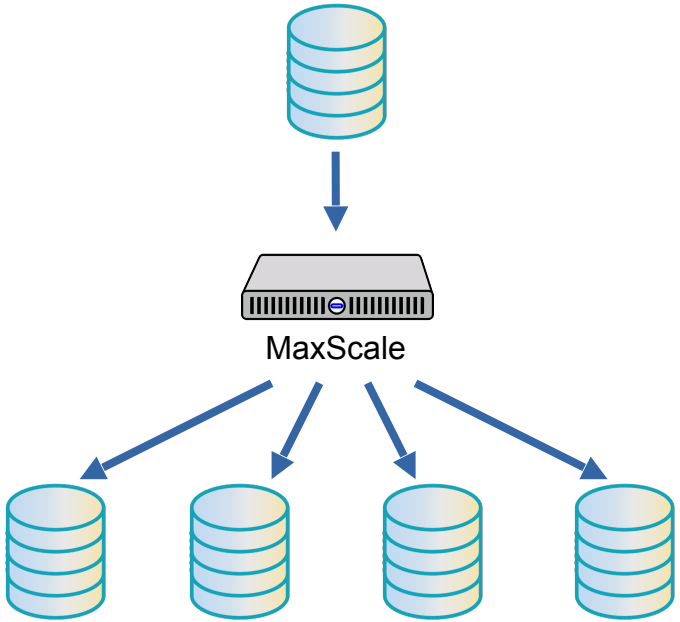
MariaDB

TOPOLOGIES

Multi-Master Topology



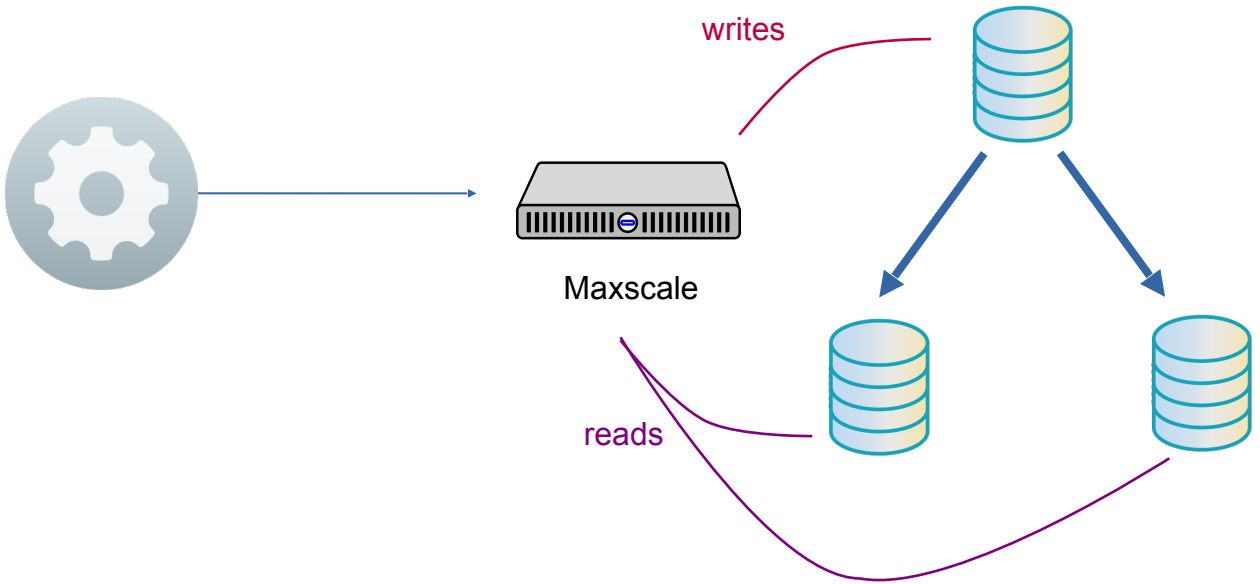
Binlog Router Topology



MariaDB

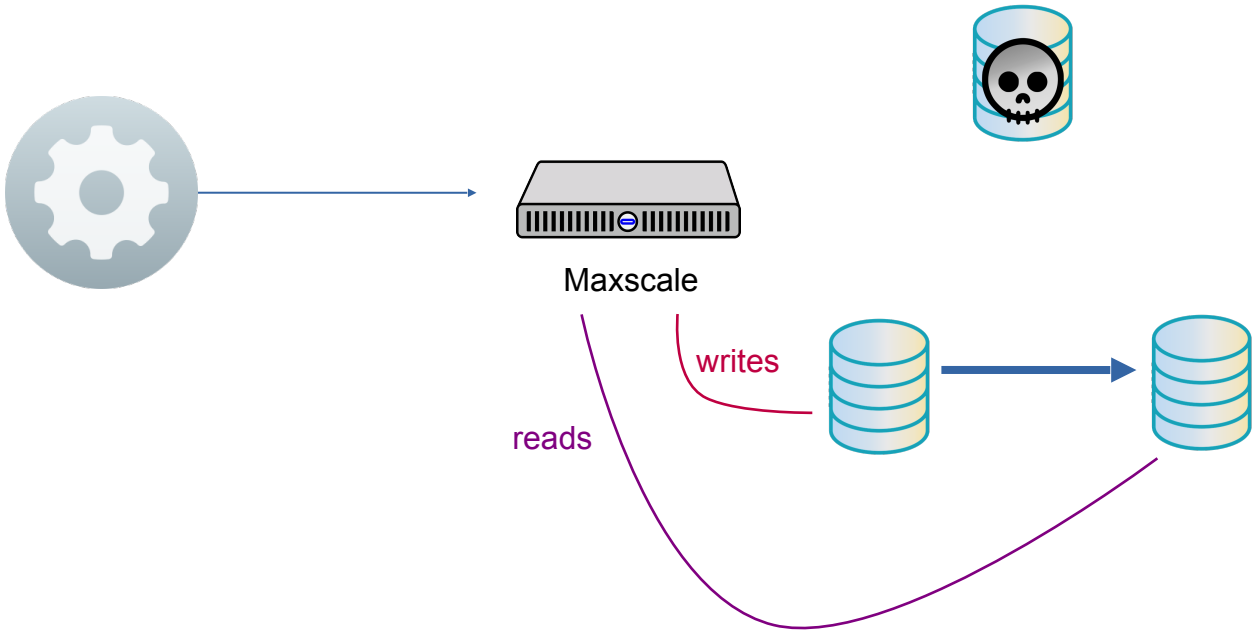
TOPOLOGIES

Maxscale Read/Write Split
Topology with failover



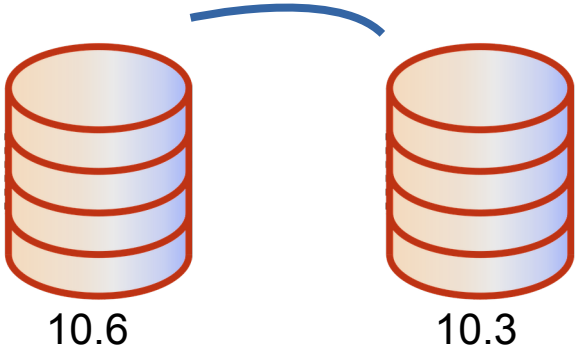
TOPOLOGIES

Maxscale Read/Write Split
Topology with failover

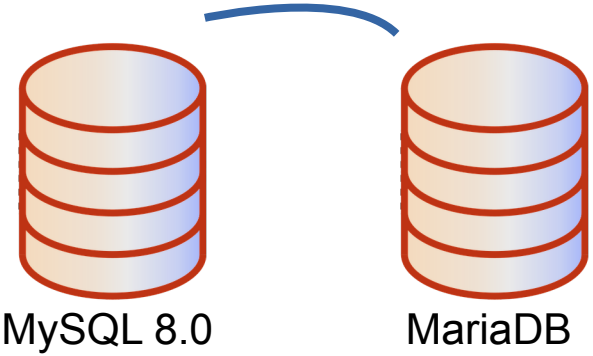


TOPOLOGIES

Although they might work, the following topologies are not supported.



Avoid replicating to an older release. You can probably get away with replicating from 10.6.x to an older 10.6.y. When possible upgrade the replica before the primary.



MariaDB Server 10.2 and later can replicate from a MySQL 5.7 primary server. Review documented incompatibilities between MySQL and Mariadb in the Mariadb knowledge base.

https://mariadb.com/kb/en/mariadb-vs-mysql-compatibility/

MariaDB

COMMANDS TO MANAGE REPLICATION

```
stop slave;
start slave;
reset slave;
reset master;
set global gtid_slave_pos='0-1-100’;
show binary logs;
show binlog events in 'mariadb-bin.005522' limit 20;
show slave hosts;  -- run on master
show master status; -- run on master
show slave status\G -- run on slaveshow global variables like '%gtid%';
```

```
mariadb-binlog /var/log/mysql/mariadb-bin.005522 | grep -i create
```

CHANGE MASTER COMMAND

```
change master to
  master_host='dran1.edw.ee',
  master_port=3306,
  master_user='dranapp_repl1',
  master_password='password',
  master_use_gtid=slave_pos;
```


MariaDB

```
change master to master_use_gtid=no
                 MASTER_LOG_FILE='mariadb-bin.000002',
                 MASTER_LOG_POS=330;
```

(Legacy)

```
change master to master_use_gtid=current_pos;
```

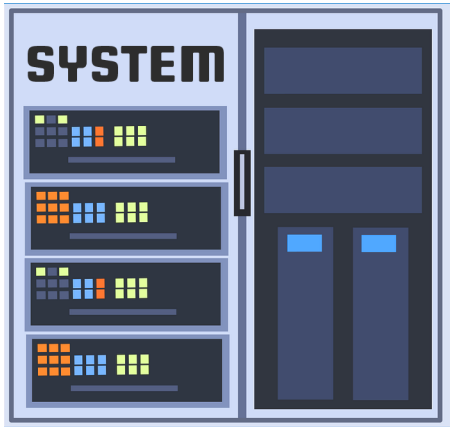
(Deprecated from MariaDB 10.10)

```
SET GLOBAL gtid_slave_pos='0-10-551';
change master to master_use_gtid=slave_pos;
```

Set the slave gtid to the desired next transaction from master -1.

MariaDB

SYSTEM VARIABLES FOR REPLICATION



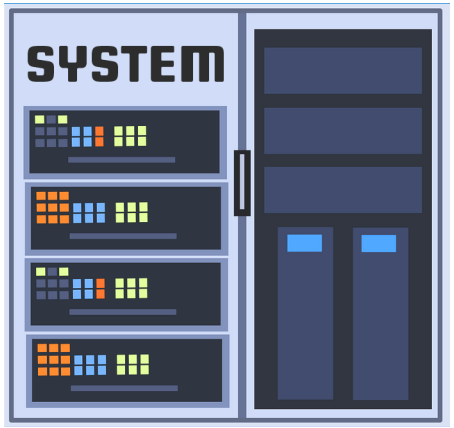
https://mariadb.com/kb/en/replication-and-binary-log-system-variables/

90 system variables related to binary logging and replication

show global variables like

binlog_format = [mixed | row | statement]
gtid_strict_mode = [off | on]
log_slave_updates = [off | on]
replicate_do_db
replicate_do_table
replicate_ignore_db
replicate_ignore_table
server_id
slave_exec_mode = [idempotent | strict]
slave_skip_errors = [list of codes | all | off]

STATUS VARIABLES FOR REPLICATION



https://mariadb.com/kb/en/replication-and-binary-log-status-variables/

40 status variables related to binary logging and replication

show global status like

slaves_connected -- on master

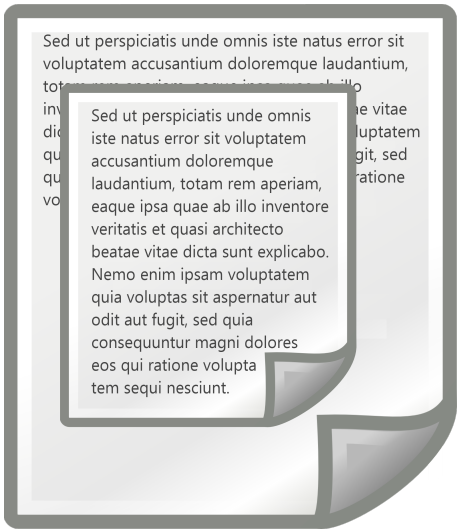
slave_running -- on slave

-- alternative method:
select VARIABLE_VALUE from information_schema.global_status where VARIABLE_NAME='Slave_running' limit 1;

KEY CONCEPT #3

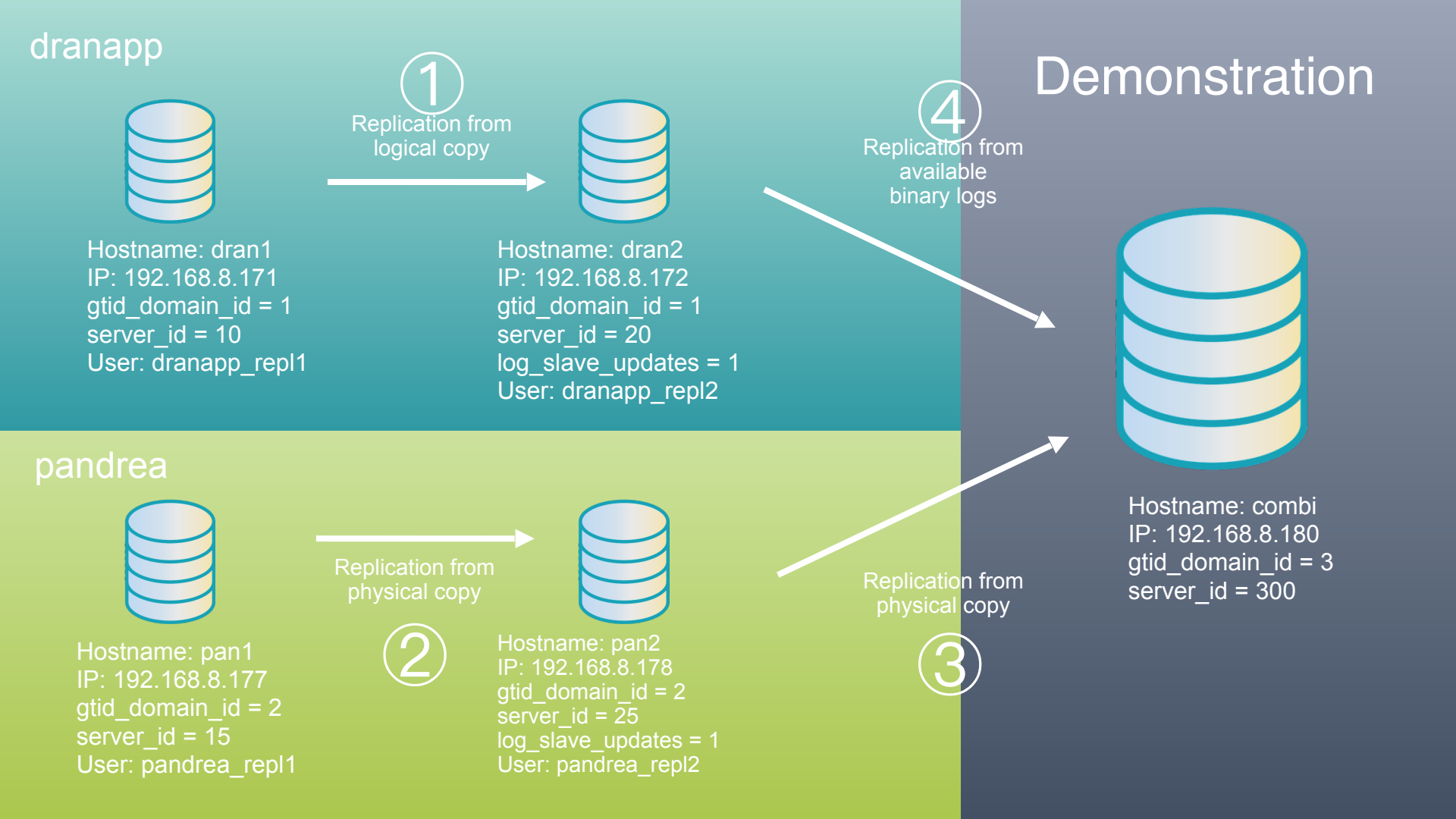
Physical copy

Logical copy

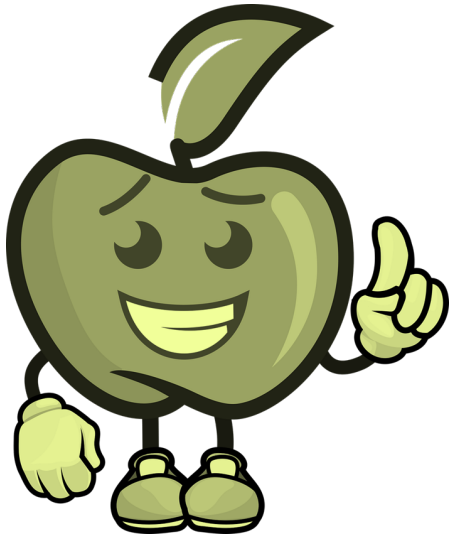


SCENARIOS FOR STARTING REPLICATION

MariaDB


dranapp
Demonstration
1
Replication from
logical copy
4
Replication from
available
binary logs
Hostname: dran1
IP: 192.168.8.171
gtid_domain_id = 1
server_id = 10
User: dranapp_repl1
Hostname: dran2
IP: 192.168.8.172
gtid_domain_id = 1
server_id = 20
log_slave_updates = 1
User: dranapp_repl2
pandrea
Replication from
physical copy
2
Replication from
physical copy
3
Hostname: combi
IP: 192.168.8.180
gtid_domain_id = 3
server_id = 300
Hostname: pan1
IP: 192.168.8.177
gtid_domain_id = 2
server_id = 15
User: pandrea_repl1
Hostname: pan2
IP: 192.168.8.178
gtid_domain_id = 2
server_id = 25
log_slave_updates = 1
User: pandrea_repl2

CHANGES ON A REPLICA/SLAVE THAT WILL NOT BREAK REPLICATION



- When possible, treat a replica server as a read-only server
- Avoid performing DML on objects you are replicating from
  - Perform DDL or DML on a schema that does not exist on the primary

WAYS TO BREAK REPLICATION FROM THE PRIMARY/MASTER

- Turn off binary logging for the session, then run DML or DDL commands
  - SET SESSION sql_log_bin = 0;
- Reset binary logging
  - reset master;
- Make any changes to gtid_seq_no for the session, then run DML or DDL commands
  - SET SESSION gtid_seq_no = 100;
- In a shutdown, binary log dump thread is killed before all client threads, and a client thread performs DDL or DML. Can be avoided with:
  - SHUTDOWN WAIT FOR ALL SLAVES;
  - set session skip_replication=1;

WAYS TO BREAK REPLICATION FROM THE PRIMARY/MASTER

- Insert into a table with an auto_increment column or insert an expected incoming primary key
- Create a new object with a name that is later created on primary
- Delete or update a row on replica that conflicts with a later delete or update on the primary
- Start replication from a GTID sequence that does not include all necessary changes, leaving replica inconsistent with primary

SCENARIOS FOR BREAKING REPLICATION

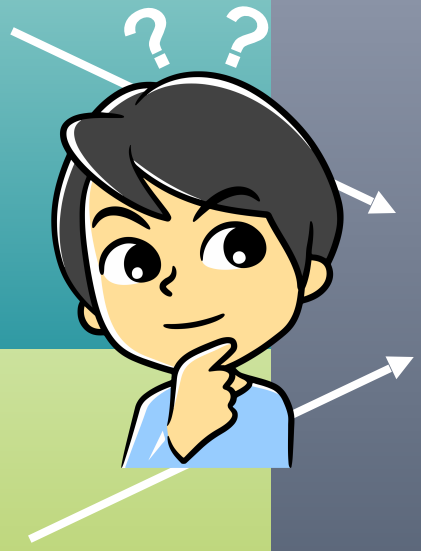
MariaDB

Demonstration

dranapp

Hostname: dran1
IP: 192.168.8.171
gtid_domain_id = 1
server_id = 10
User: dranapp_repl1

Hostname: dran2
IP: 192.168.8.172
gtid_domain_id = 1
server_id = 20
log_slave_updates = 1
User: dranapp_repl2



Hostname: combi
IP: 192.168.8.180
gtid_domain_id = 3
server_id = 300

pandrea

Hostname: pan1
IP: 192.168.8.177
gtid_domain_id = 2
server_id = 15
User: pandrea_repl1

Hostname: pan2
IP: 192.168.8.178
gtid_domain_id = 2
server_id = 25
log_slave_updates = 1
User: pandrea_repl2

REPLICATION REVIEW

What We Learned

- Replication is a process of copying transactions in real-time from a primary database to a replica database
- Avoid conflicts among servers by using different gtid_domain_id's
- Binary logging must be enabled on the primary to copy transactions to a replica
- Various topologies can be created with replication
- Replication is a logical copy of data
- Replication is set up with a user account connection established from the replica to the primary
- Replication can break and you should be prepared to fix it


MariaDB
OPEN WORKS
THANK YOU


MariaDB
OPEN WORKS
BE UNSTOPPABLE