The Exciting MySQL 5.7 Replication Enhancements

Luís Soares (luis.soares@oracle.com)
Principal Software Engineer, MySQL Replication Team Lead

 Copyright © 2016, Oracle and/or its affiliates. All rights reserved. |

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes 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 upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Program Agenda

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Program Agenda

1. Background
2. MySQL 5.7 Replication Generally Available
3. The New Replication Features in MySQL 5.7
4. Development Sneak Peek
5. Roadmap
6. Summary

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

1 MySQL 5.7 Replication Generally Available

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Interesting Stats Related to MySQL 5.7

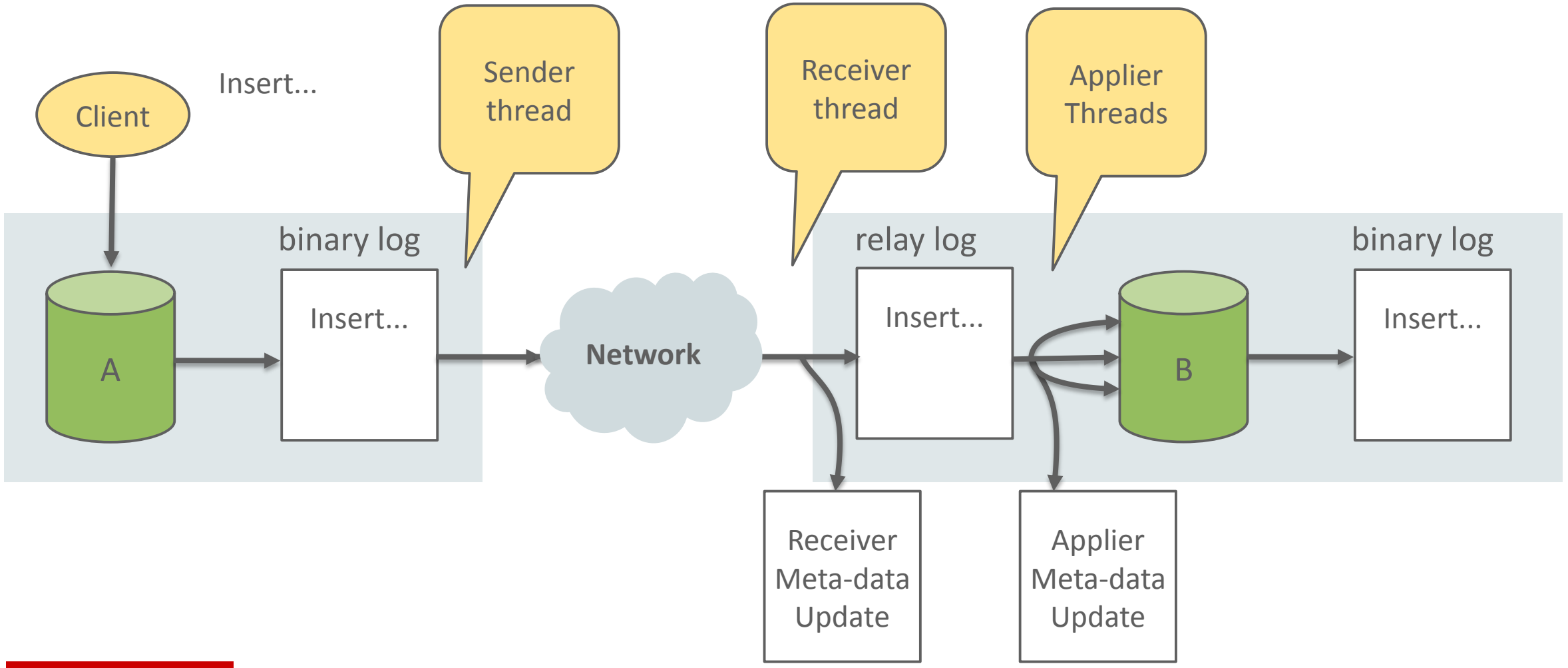
- **40 replication worklogs** pushed to MySQL 5.7
  - 29 related to on MySQL replication Core
  - 11 are ground work for MySQL Group Replication.
- **8 contributions** merged into MySQL 5.7.
- **19 major enhancements** to the replication core.
- **14 refactoring and modularization** related changes to the code.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

2 Background

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

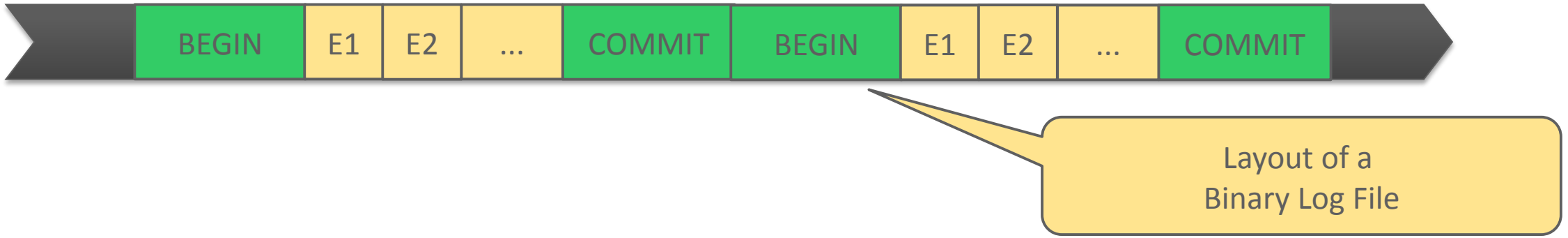
Background: Replication Components



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Background: Replication Components

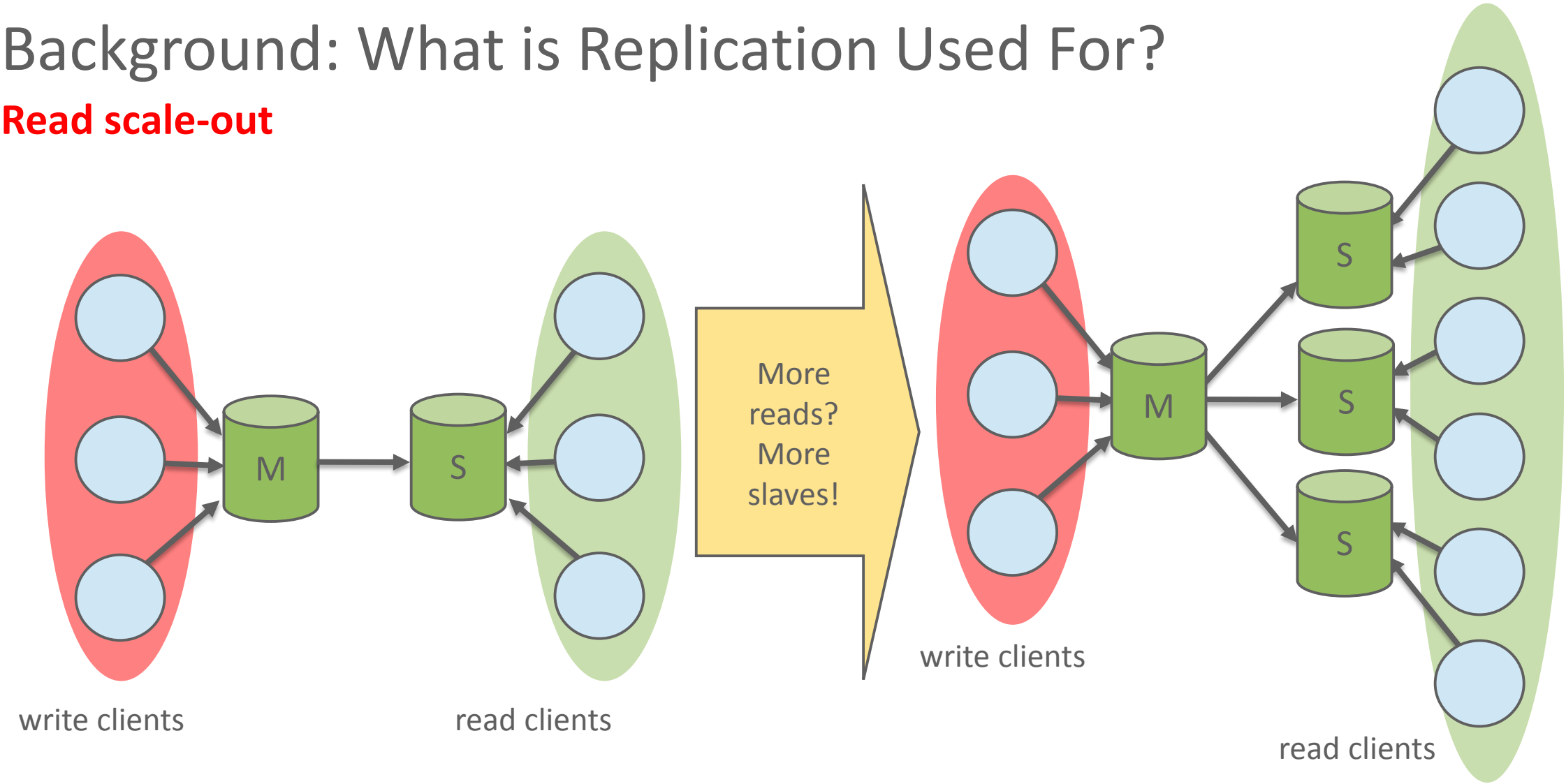
- **Binary Log**
  - File based logical log that records the changes on the master.
  - Statement or Row based format (may be intermixed).
  - Each transaction is split into groups of events.
  - Control events: Rotate, Format Description, Gtid, ...



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Background: What is Replication Used For?

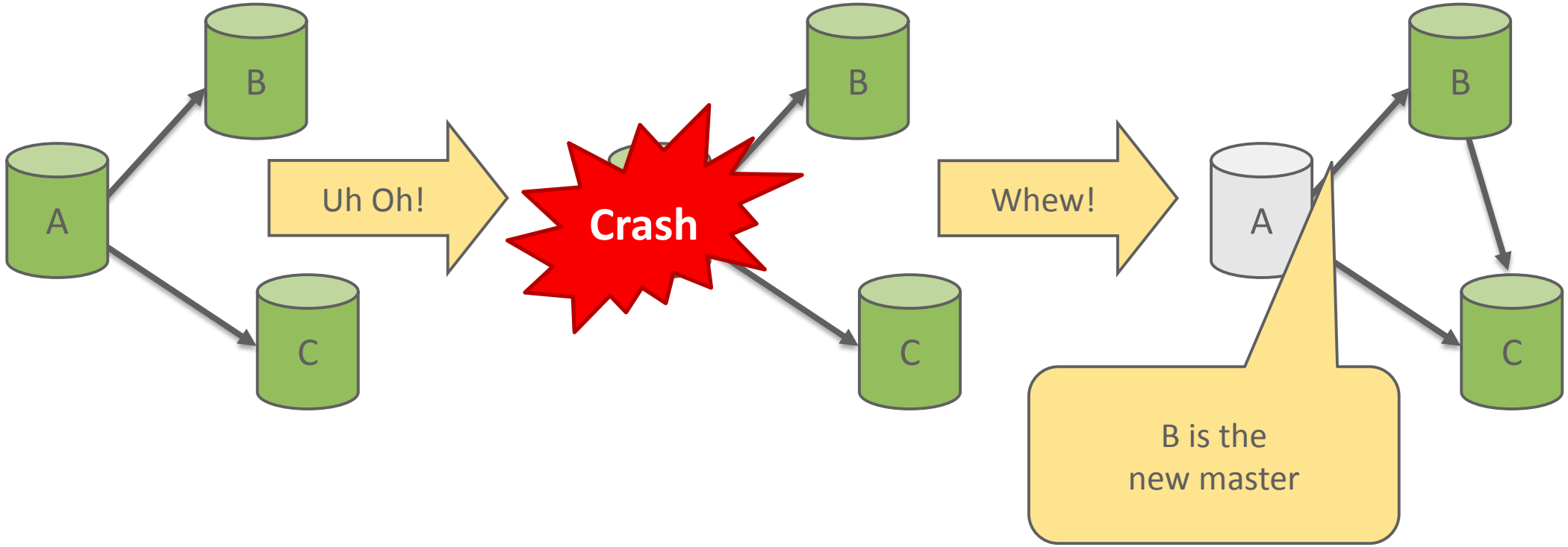
Read scale-out



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Background: What is Replication Used For?

Redundancy: If master crashes, promote slave to master



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Background: What is Replication Used For?

On-line Backup and Reporting

write clients

business intelligent client applications
reporting client applications
big queries client applications

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Background: What is Replication Used For?

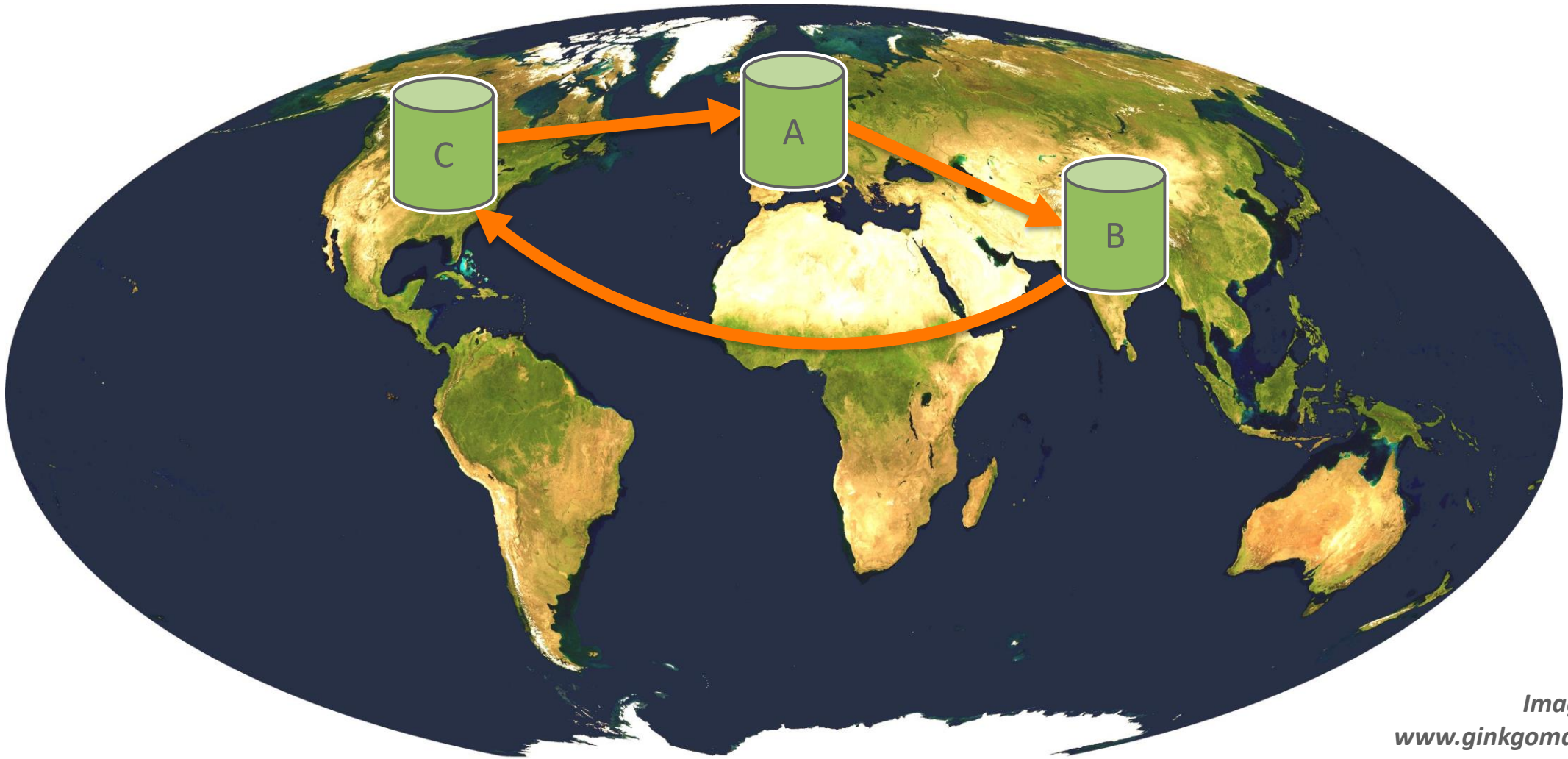


Image from
www.ginkgomaps.com

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

3 The New Replication Features in MySQL 5.7

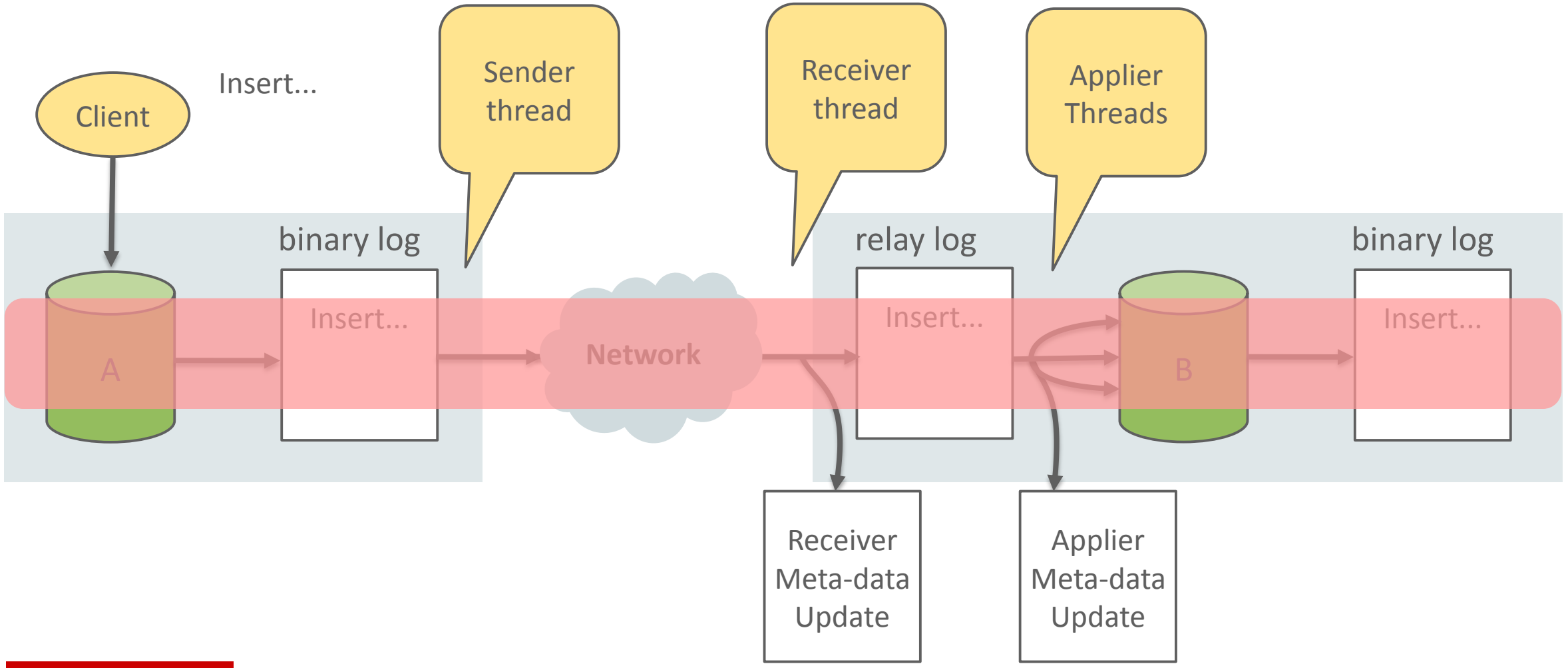
Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

3 The New Replication Features in MySQL 5.7

3.1 Usability / Online

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Online Reconfiguration of Global Transaction Identifiers



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Online Reconfiguration of Global Transaction Identifiers

- The procedure is online.
  - Both reads and writes are allowed while global transaction identifiers are being turned on or off in the entire replication cluster.
- No need to synchronize servers, ever.
- No need to restart servers.
- No need to change replication topology.
  - Works in arbitrary topologies.
- Should anything happen in the middle of the procedure, you can always roll back.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Online Reconfiguration of Global Transaction Identifiers

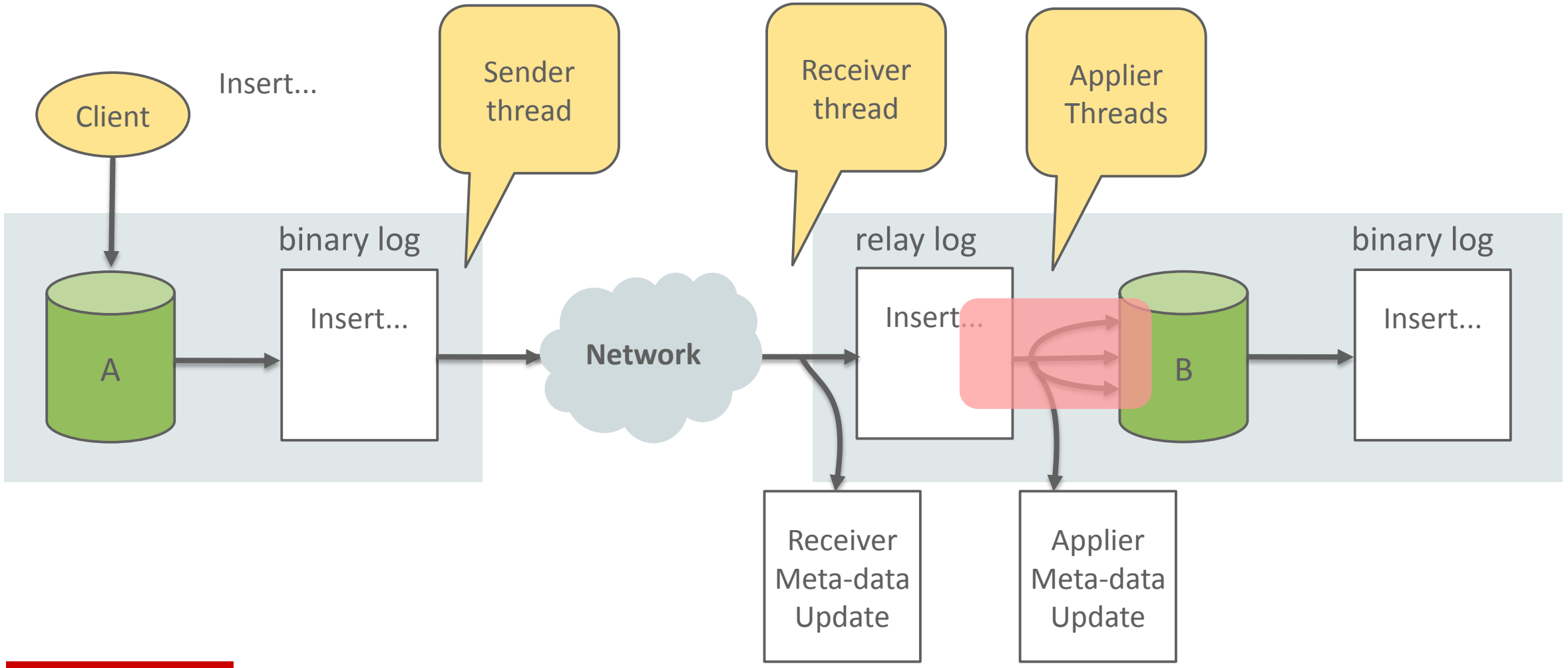
- Over simplified procedure to turn on global transaction identifiers:

1) SET @@GLOBAL.GTID_MODE = OFF_PERMISSIVE; (on every server)
2) SET @@GLOBAL.GTID_MODE = ON_PERMISSIVE; (on every server)
3) wait for a bit longer than your replication lag
4) SET @@GLOBAL.GTID_MODE = ON; (on every server)

- Details: http://dev.mysql.com/doc/refman/5.7/en/replication-mode-change-online-enable-gtids.html

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Online Reconfiguration of Replication Filters



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

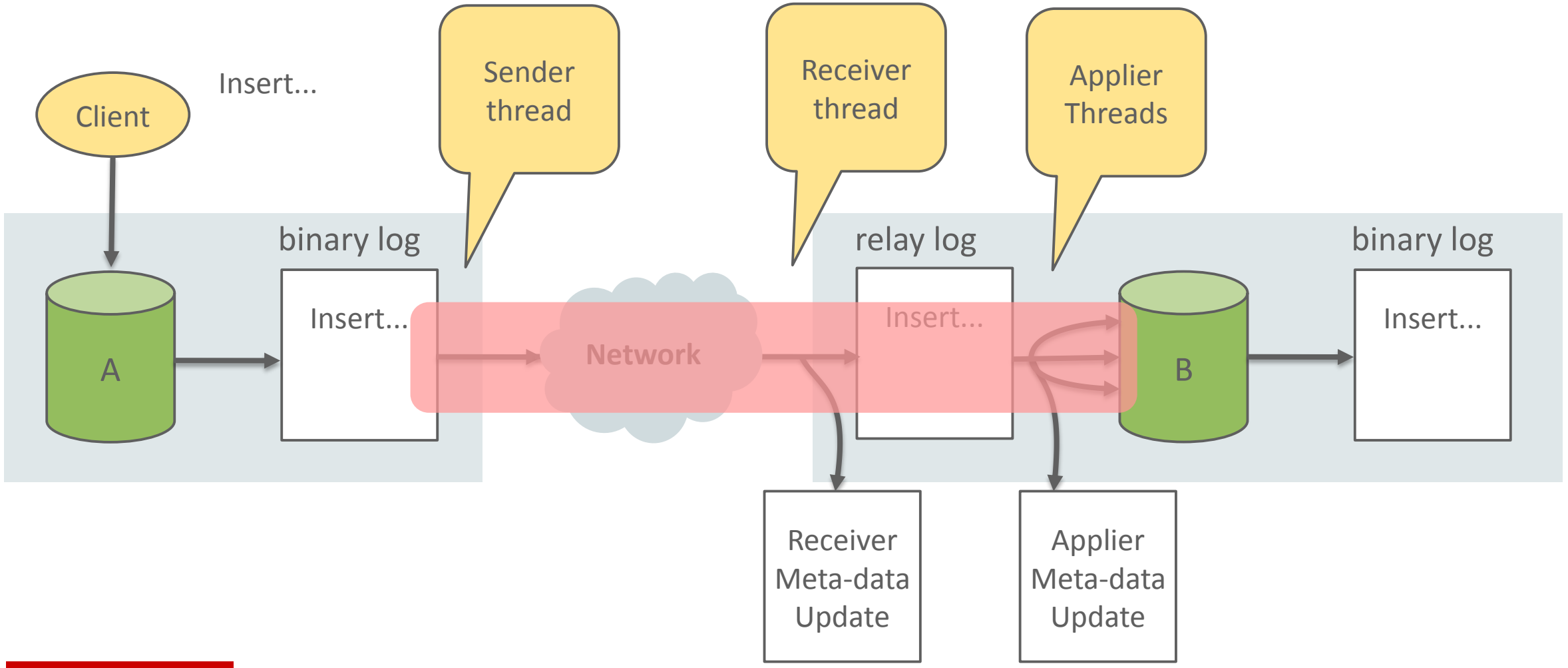
Online Reconfiguration of Replication Filters

- Change Slave's Replication Filters dynamically.
  - No need to stop and restart the slave server for setting new replication filtering rules.
  - All slave filters are supported.
  - Values can be input in various character sets.

```
mysql> CHANGE REPLICATION FILTER REPLICATE_DO_DB= (db1, db2)
```


Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

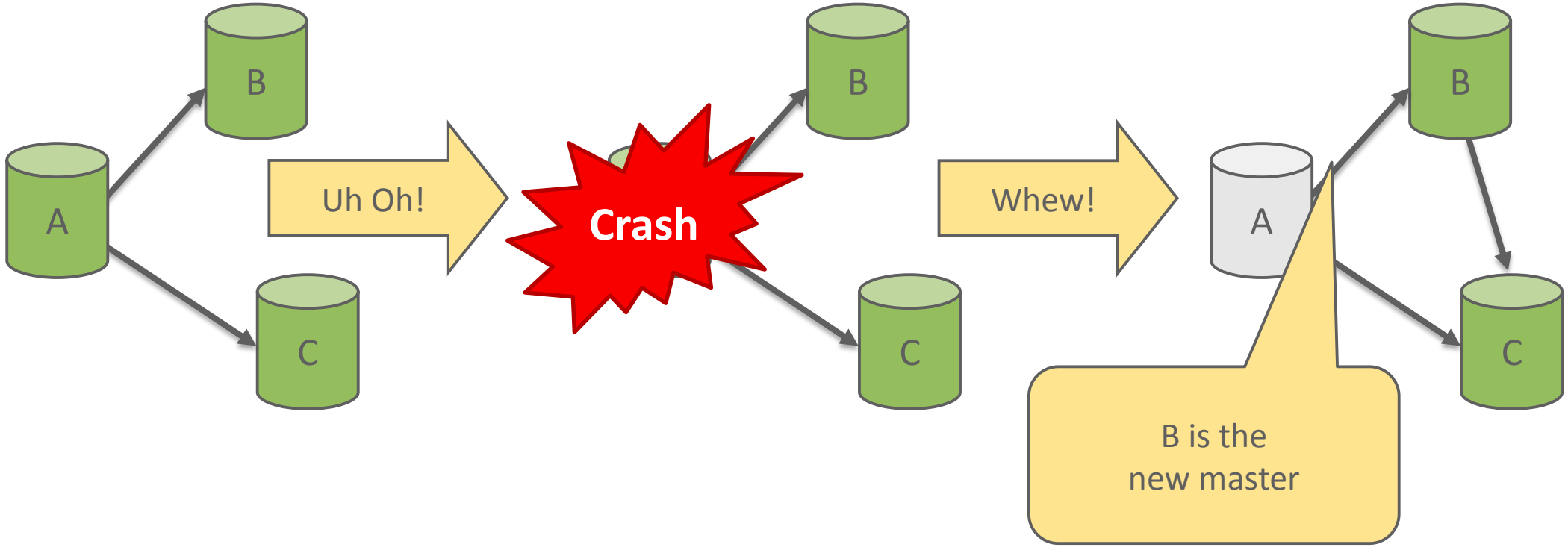
Online Reconfiguration of Replication Receiver/Applier



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Online Reconfiguration of Replication Receiver/Applier

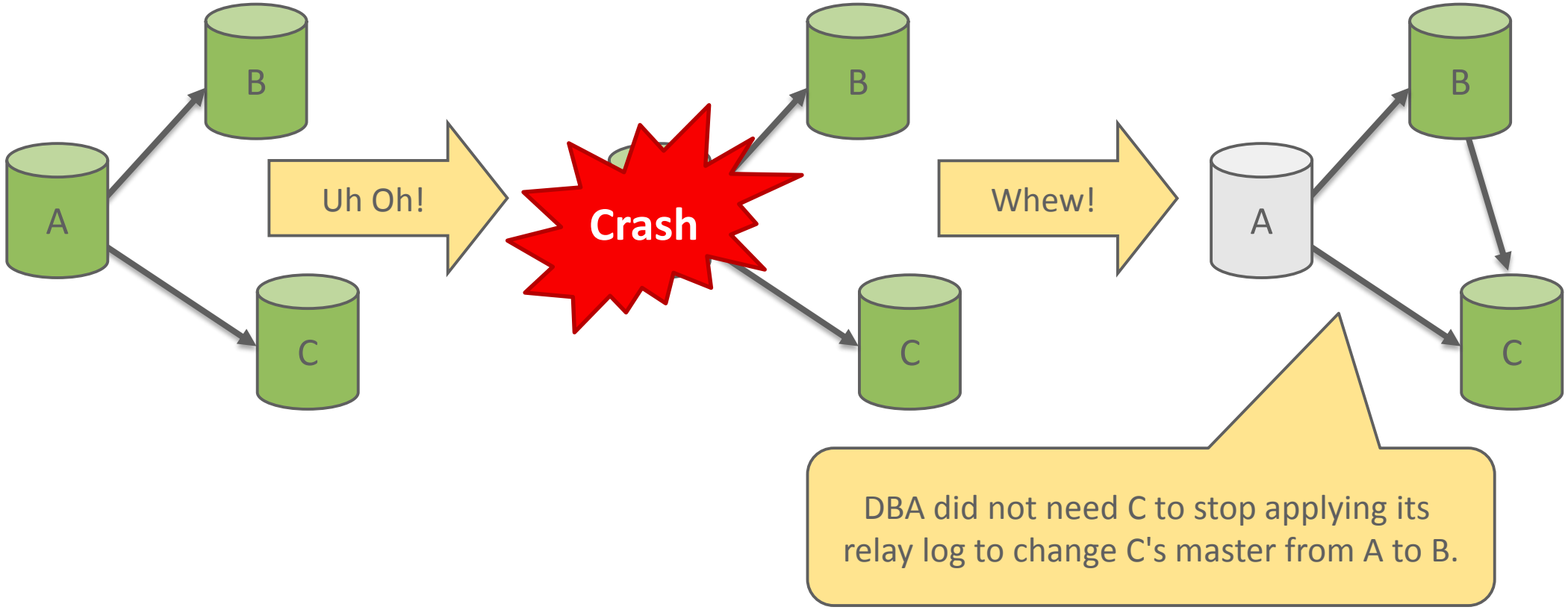
Change master from A to B without stopping the applier threads.



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Online Reconfiguration of Replication Receiver/Applier

Change master from A to B without stopping the applier threads.



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Online Reconfiguration of Replication Receiver/Applier

- Enables more online operations during fail-over:

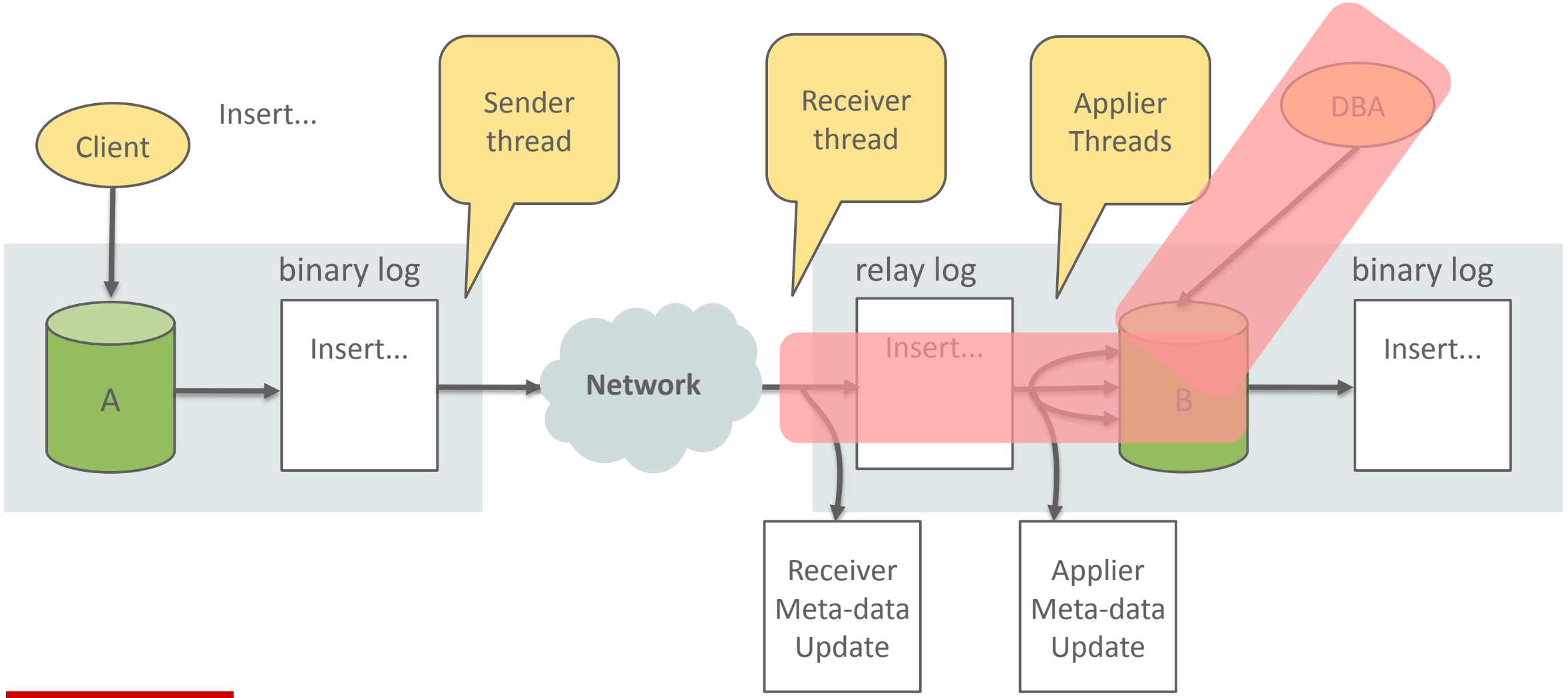
```
mysql> STOP SLAVE IO_THREAD;
mysql> CHANGE MASTER TO MASTER_HOST='master2', …;
mysql> START SLAVE IO_THREAD;
```

- Stopping, changing master and restarting the receiver thread are all done while the applier thread is running.
- Change applier properties while the receiver thread is working:

```
mysql> STOP SLAVE SQL_THREAD;
mysql> CHANGE MASTER TO MASTER_DELAY=3600, …;
mysql> START SLAVE SQL_THREAD;
```


Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Improved Replication Monitoring



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

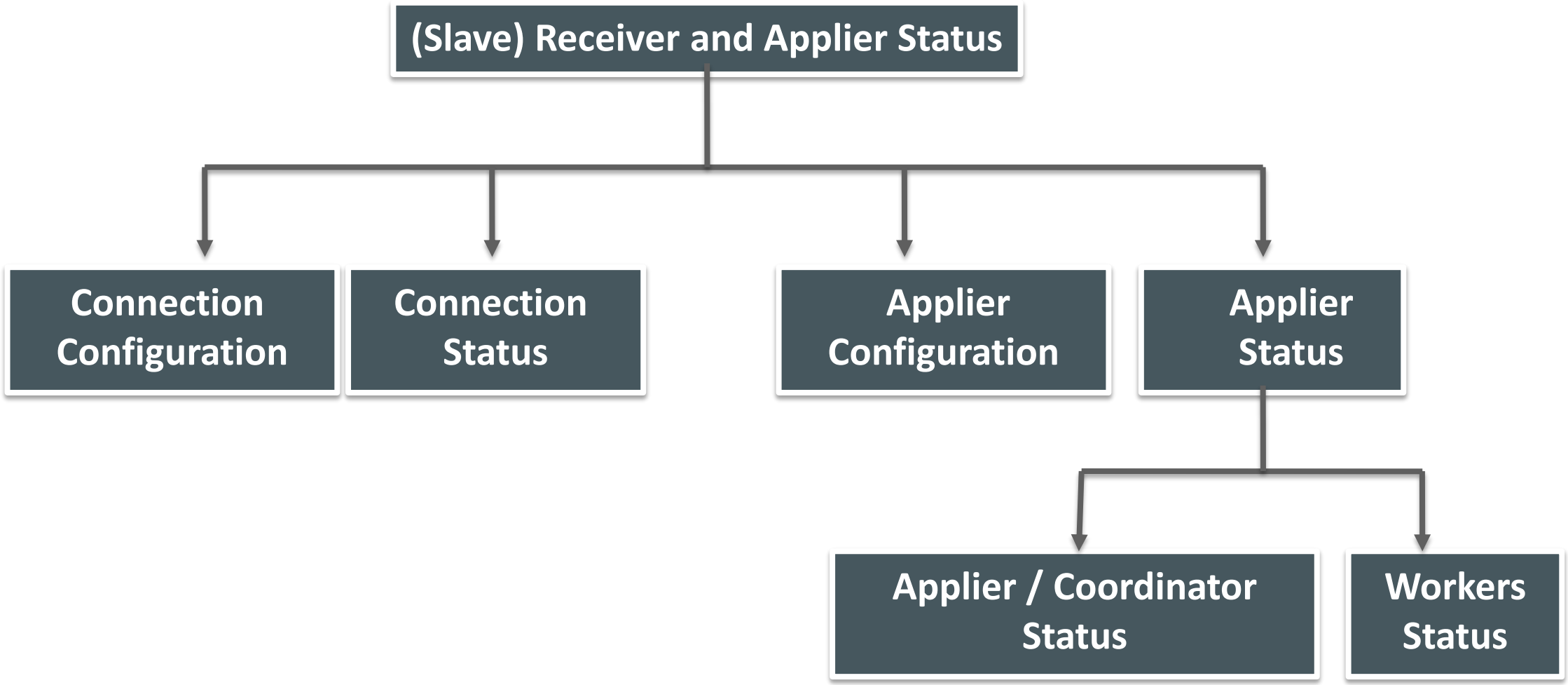
Improved Replication Monitoring

Performance Schema Replication Tables

- Monitoring through SQL.
- Grouping logically related information.
- Extensible, can be adapted to new features.
- More accurate and consistent identifier names.
- Master-slave, Multi-source, Group Replication support.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Improved Replication Monitoring



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Improved Replication Monitoring

```
mysql> select * from performance_schema.replication_applier_status_by_worker\G
*************************** 1. row ***************************
        CHANNEL_NAME:
           WORKER_ID: 1
           THREAD_ID: 35
       SERVICE_STATE: ON
LAST_SEEN_TRANSACTION: 4ba0eb86-63c3-11e4-92ba-28b2bd168d07:2368
    LAST_ERROR_NUMBER: 0
   LAST_ERROR_MESSAGE:
 LAST_ERROR_TIMESTAMP: 0000-00-00 00:00:00
*************************** 2. row ***************************
        CHANNEL_NAME:
           WORKER_ID: 2
           THREAD_ID: 36
       SERVICE_STATE: ON
LAST_SEEN_TRANSACTION: 4ba0eb86-63c3-11e4-92ba-28b2bd168d07:2367
    LAST_ERROR_NUMBER: 0
   LAST_ERROR_MESSAGE:
 LAST_ERROR_TIMESTAMP: 0000-00-00 00:00:00
2 rows in set (0,00 sec)
```


Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

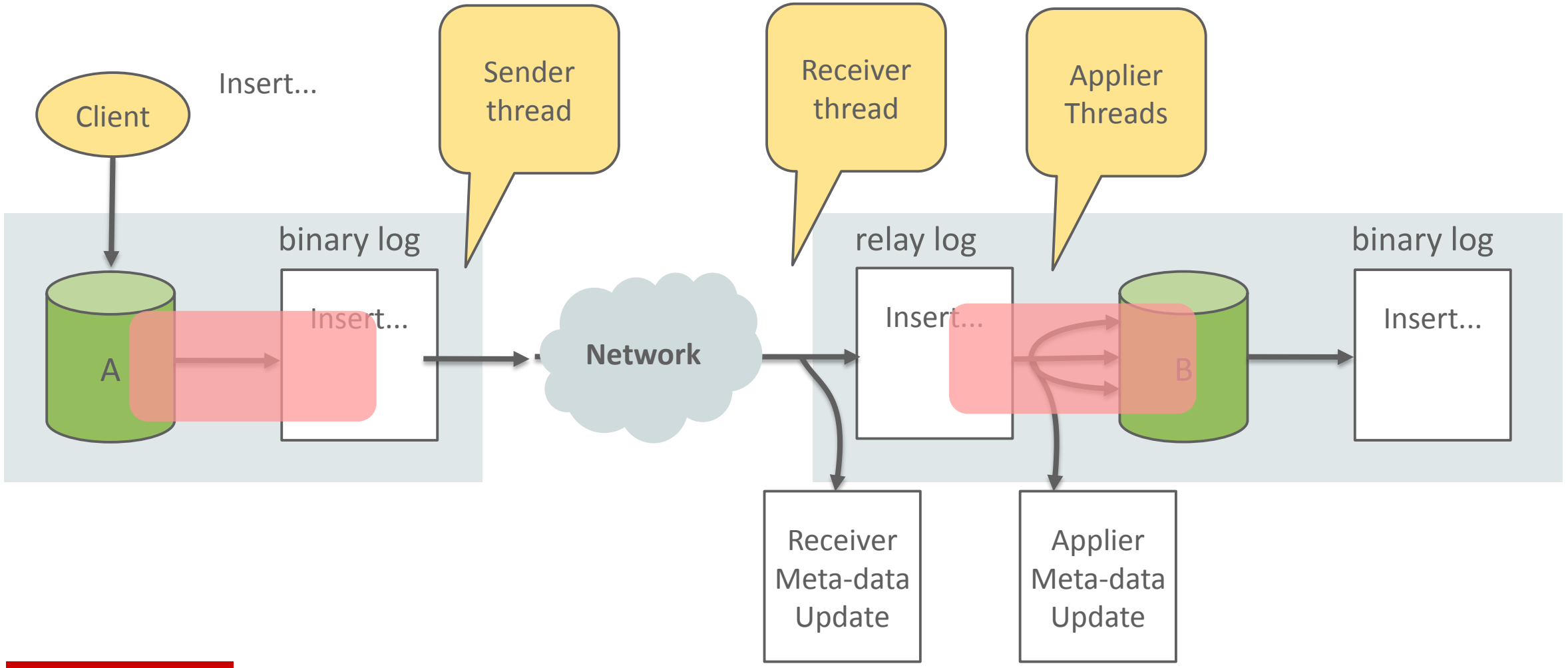
3 The New Replication Features in MySQL 5.7

3.1 Usability / Online

3.2 Performance

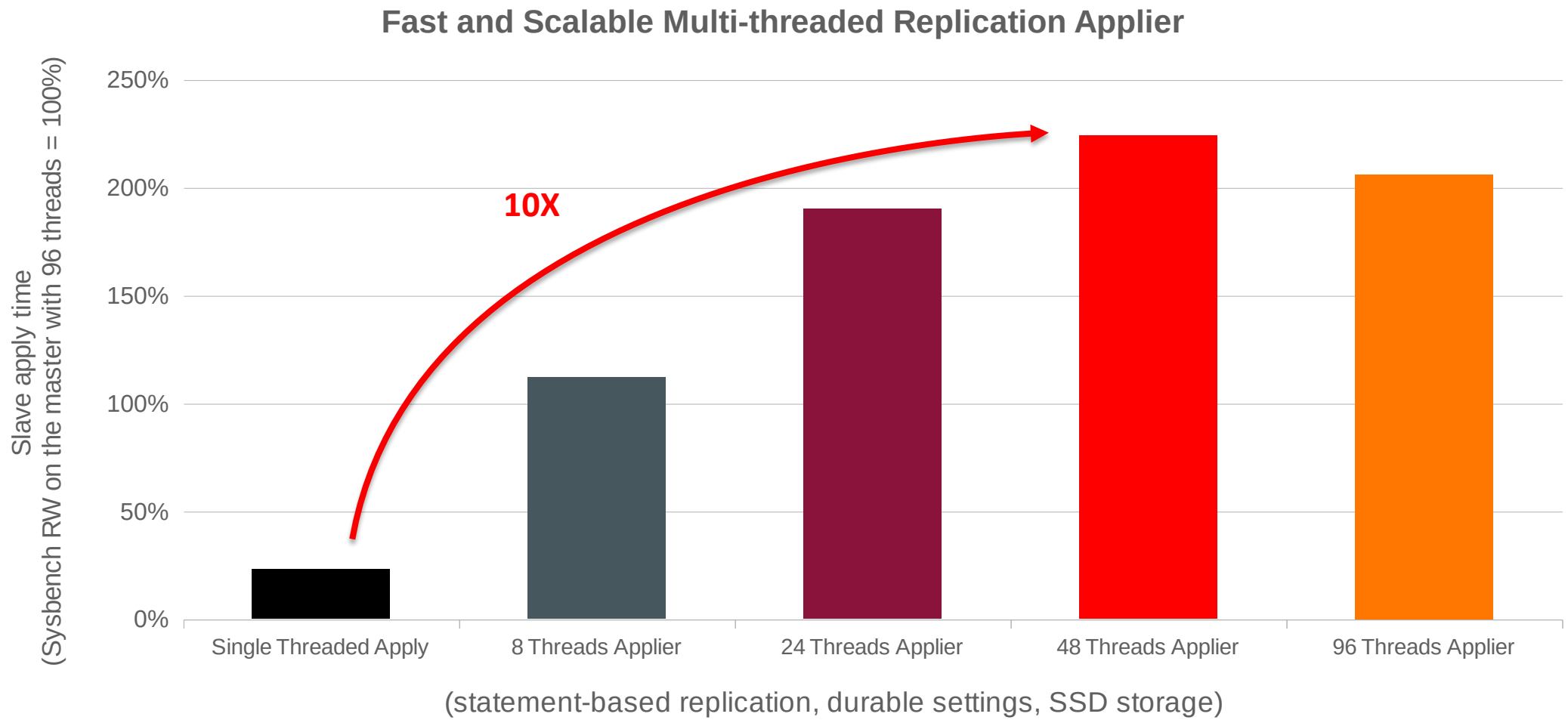
Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Improved Applier Throughput – Locking-based Parallelism



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Improved Applier Throughput – Locking-based Parallelism



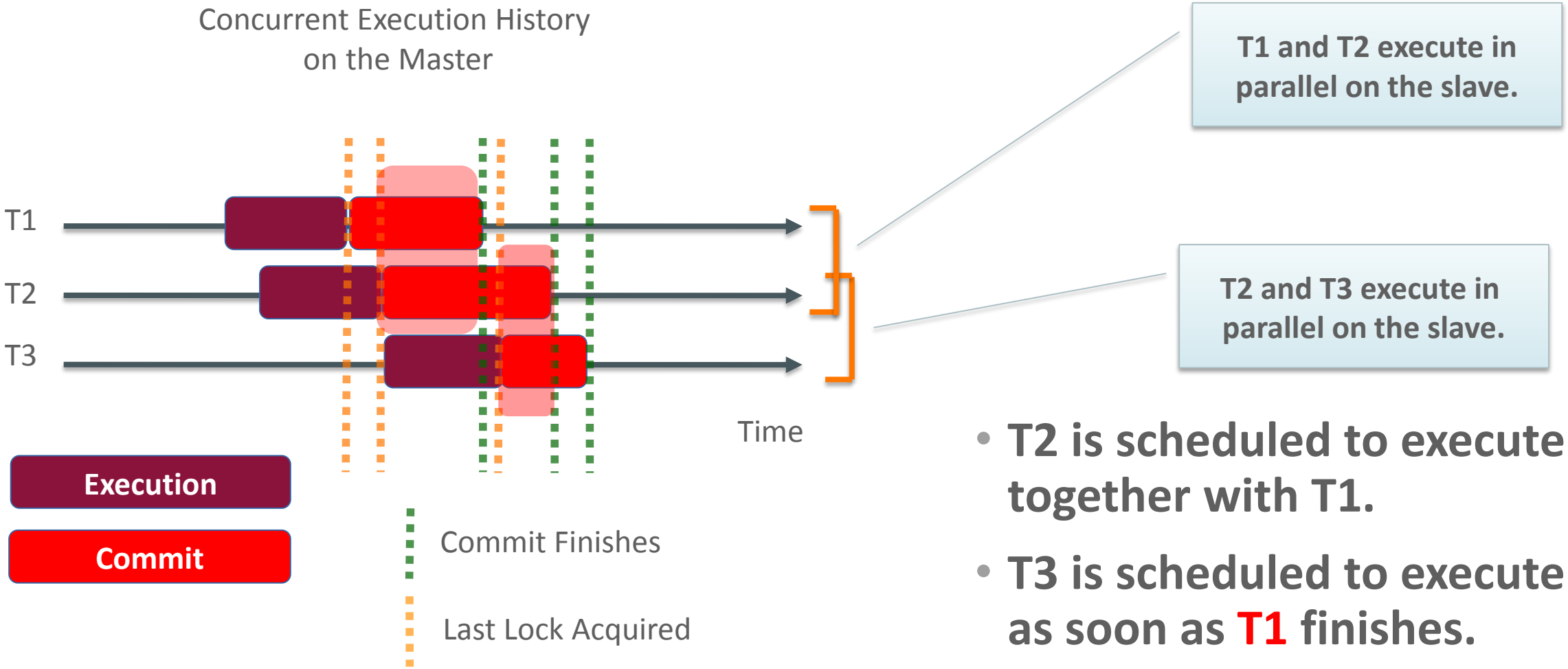
Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Improved Applier Throughput – Locking-based Parallelism

- Leverage parallelization information from master execution:
  - Concurrent transactions, which **have not blocked each other** on the master, are marked as non-contending in the binary log.
- Meanwhile, at the slave:
  - Non-contending transactions are scheduled in **parallel**;
  - Concurrent transactions commit independently, thus **no waiting** involved.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Improved Applier Throughput – Locking-based Parallelism



- **T2 is scheduled to execute together with T1.**
- **T3 is scheduled to execute as soon as T1 finishes.**

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Improved Applier Throughput – Locking-based Parallelism

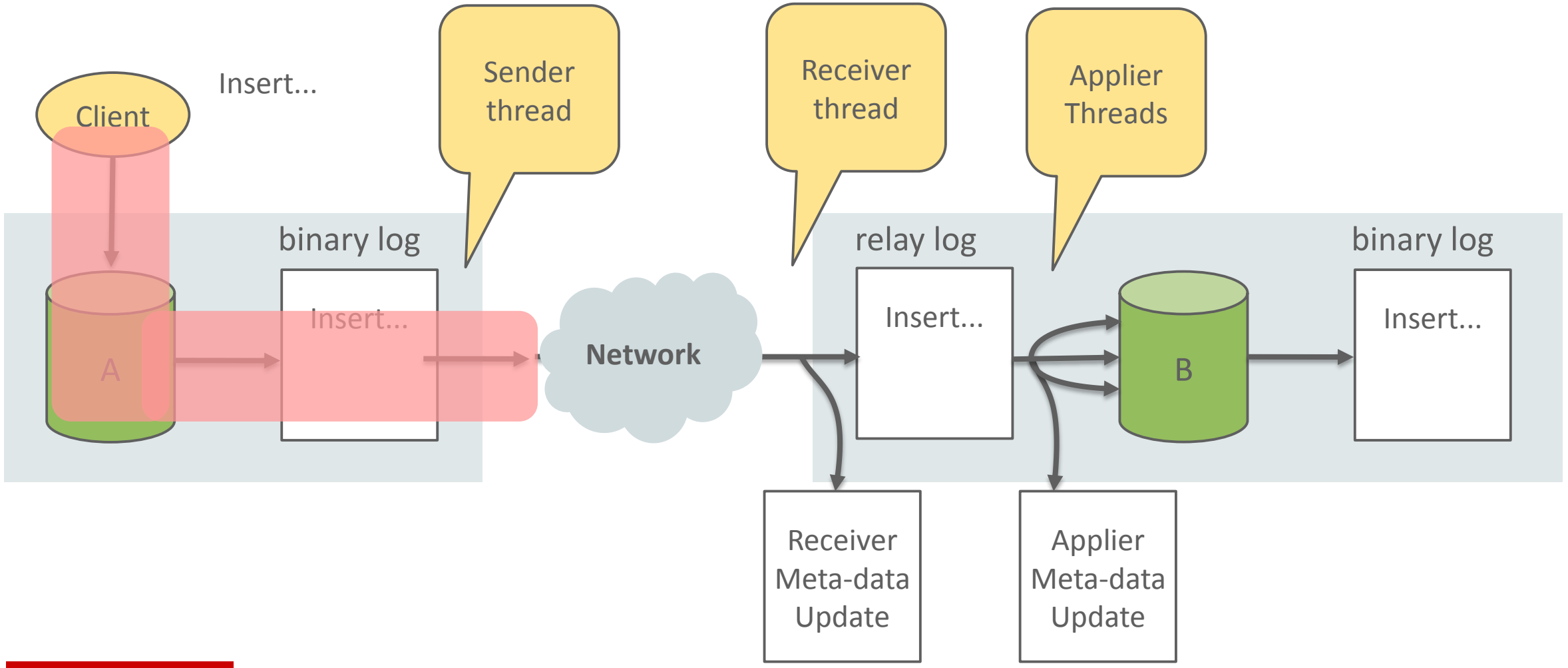
- Supports statement-based or row-based formats.
- Scheduling policy controlled through:

```
mysql> SET slave_parallel_type= [logical_clock|database]
```

  - ***logical_clock*** - means schedule based on the locking interval timestamps.
  - ***database*** - the scheduling policy from 5.6 (concurrency control done per database).
- Work to improve slave scalability continues, does not stop here.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

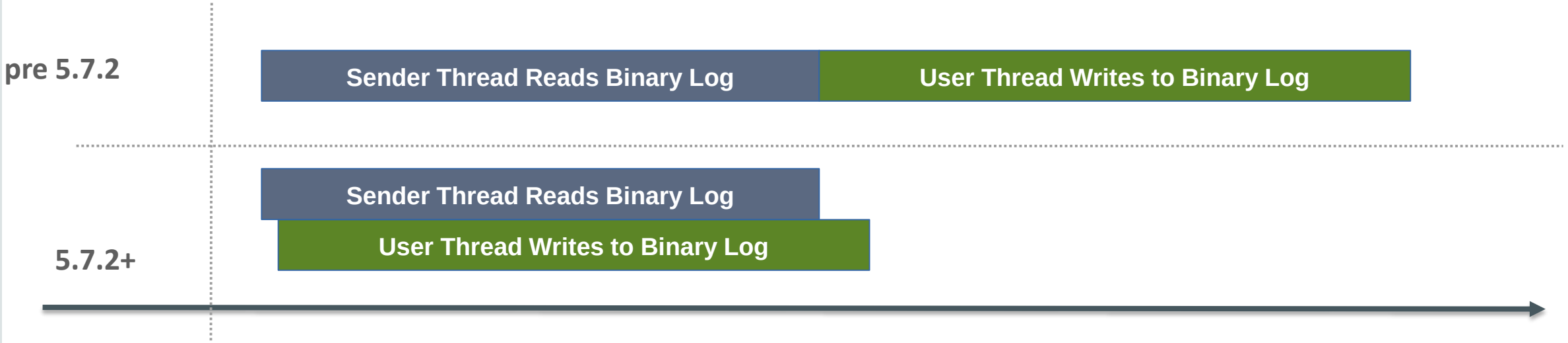
Improved User Threads-Sender Threads Synchronization



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

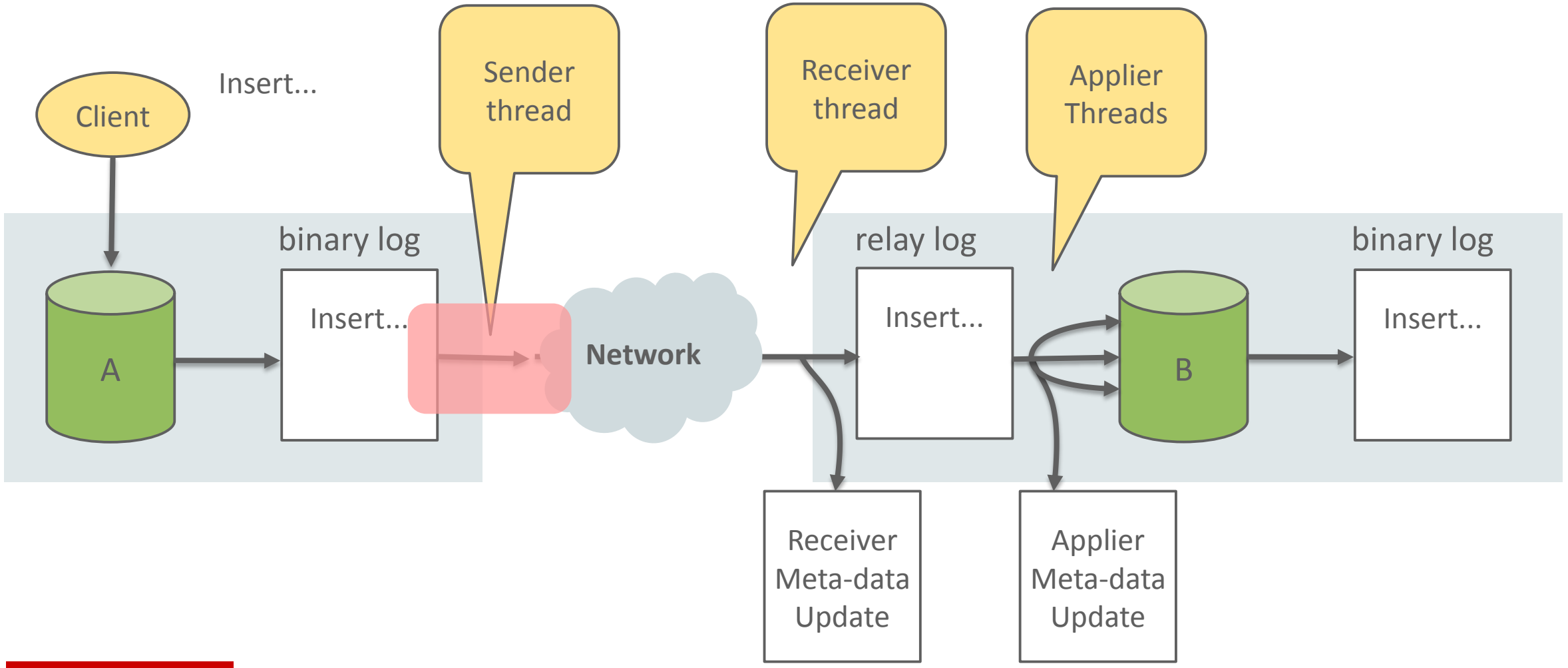
Improved User Threads-Sender Threads Synchronization

- Concurrent reads by the sender thread with ongoing writes from user threads.
  - Sender thread does not block user sessions more than necessary.
  - Higher throughput for both sender threads and user sessions.



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Faster Sender Thread



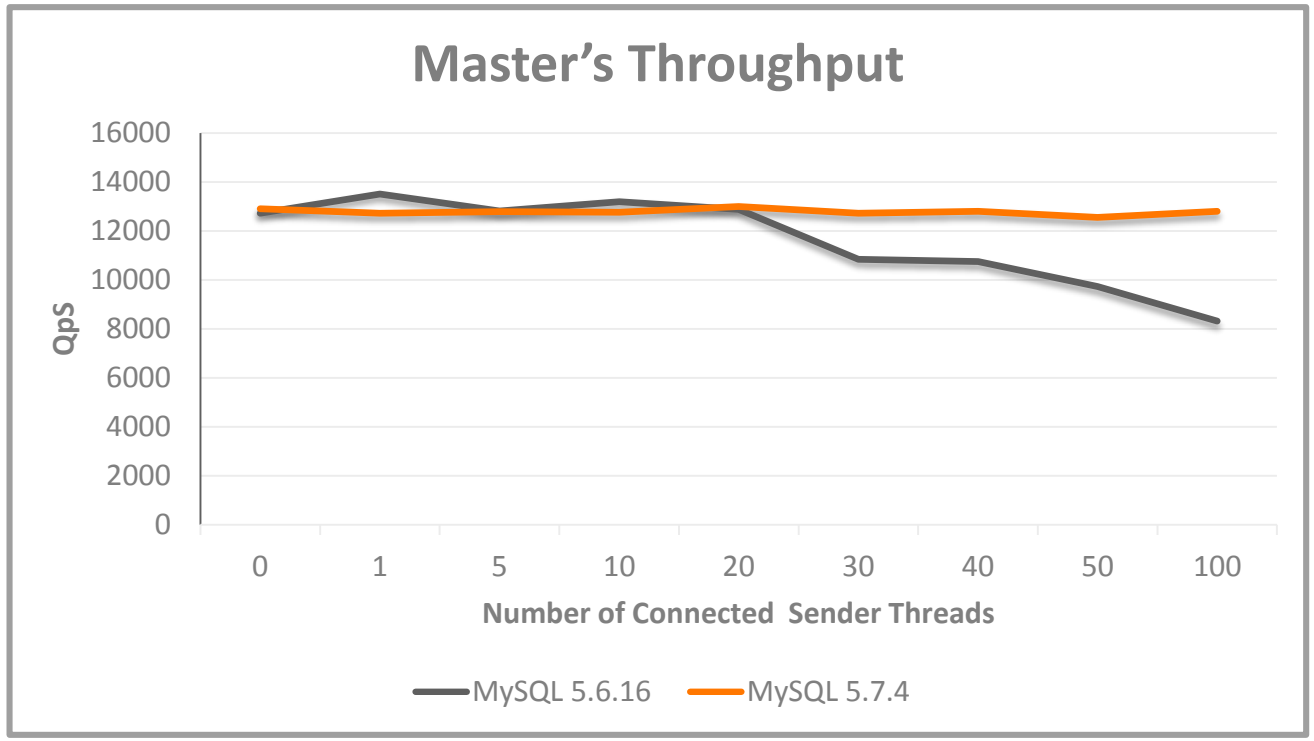
Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Faster Sender Thread

- Send buffer is not allocated and freed every time an event is sent.
- When sender threads require larger send buffer - buffer grows as needed.
- When buffer is larger than needed - buffer shrinks dynamically.
- Together with the sender thread enhancements released on MySQL 5.7.2:
  - increases master scalability;
  - reduces resource consumption (CPU);
  - Master, with dump threads attached, copes better with peak loads.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

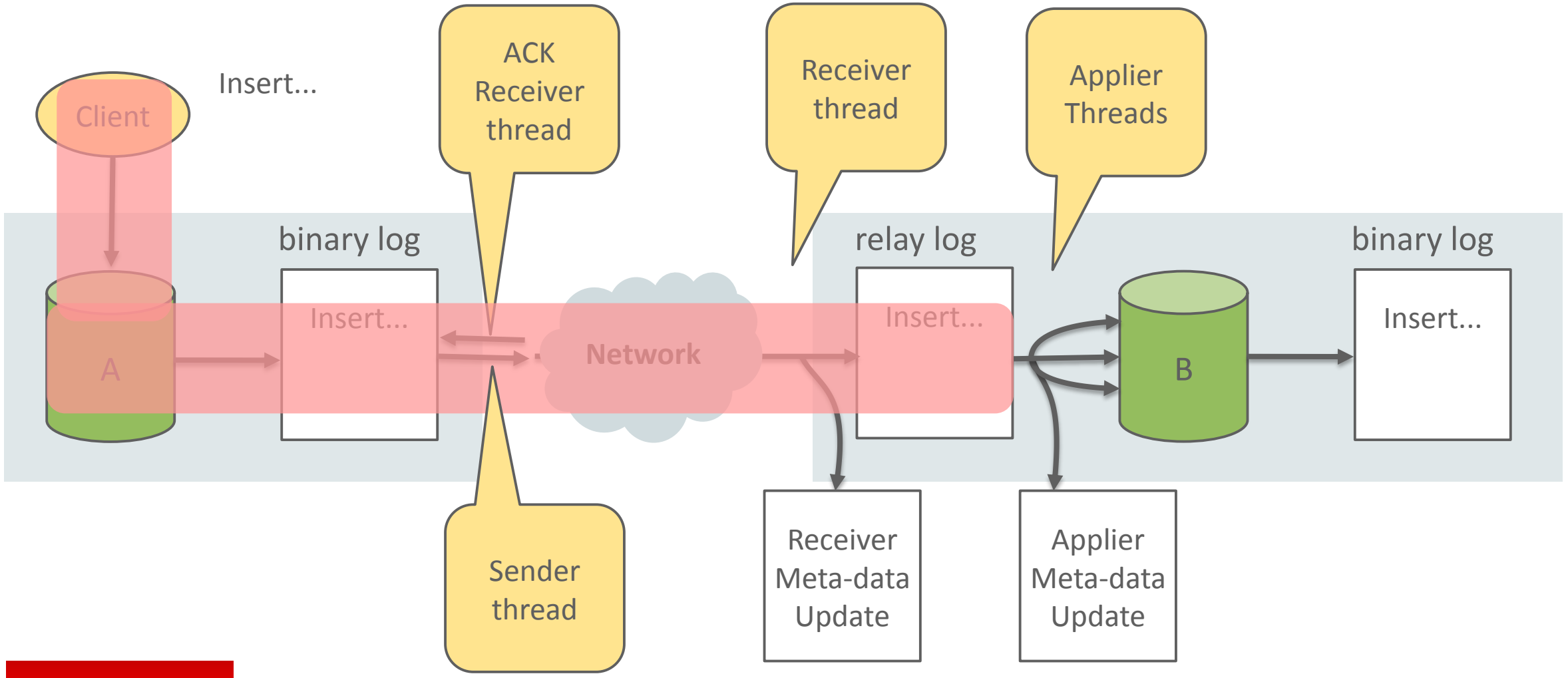
Faster Sender Thread



- Ad hoc microbenchmark using mysqlslap
- 1M queries, concurrency=200, commit=1
- N slaves attached (fake slaves using remote mysqlbinlogs)
- 48 cores HT / 512 GB RAM / HDD

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Faster Semi-sync Replication – ACK Receiver Thread



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Faster Semi-sync Replication – ACK Receiver Thread

- Consecutive transactions do not block each other while waiting for ACKs
  - Transaction t1 and t2 are sent immediately to the slave by the sender thread
  - ACKs are received only by a special thread
  - Transaction t2 does not include t1 round trip in its semi-sync overall latency
- Thread starts when semi-sync is activated

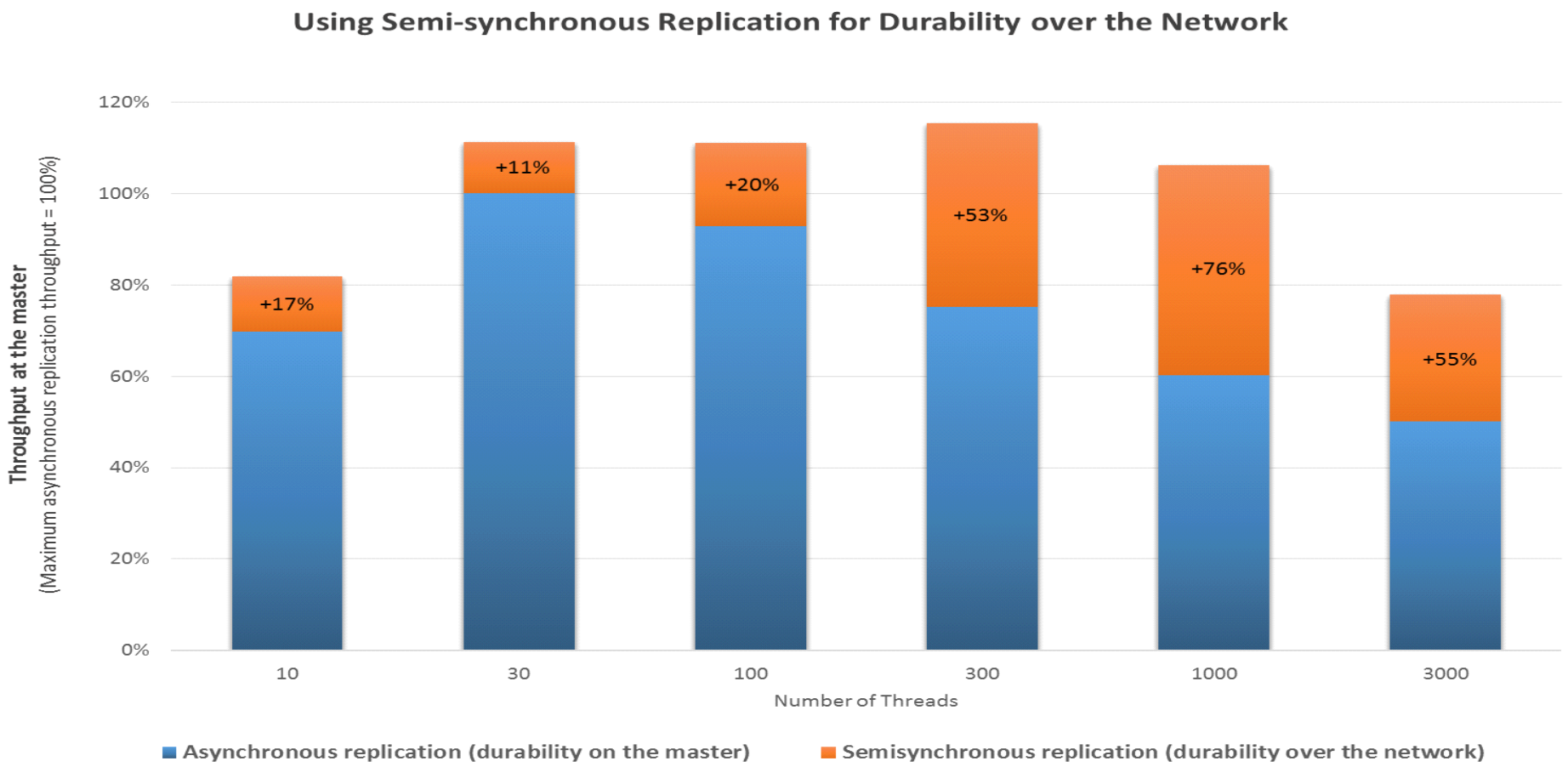
```
mysql> SET GLOBAL rpl_semi_sync_master_enabled= ON
```

- Thread stops when semi-sync is deactivated

```
mysql> SET GLOBAL rpl_semi_sync_master_enabled= OFF
```


Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Faster Semi-sync: Durability over the Network



Orange bar **percentage** shows the throughput gain (orange) when compared to the corresponding blue bar. Blue and stacked bars show throughput ratio computed against the best blue bar.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

3 The New Replication Features in MySQL 5.7

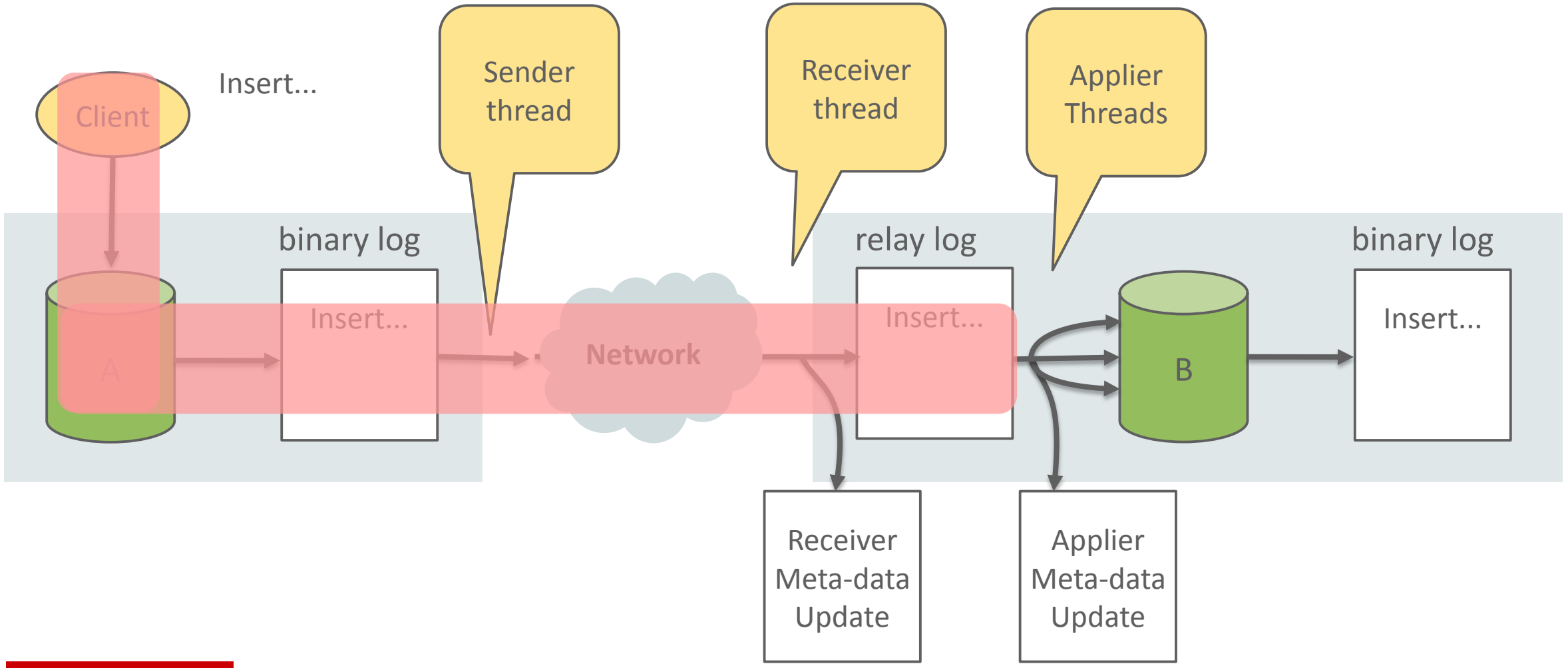
3.1 Usability / Online

3.2 Performance

3.3 Dependability

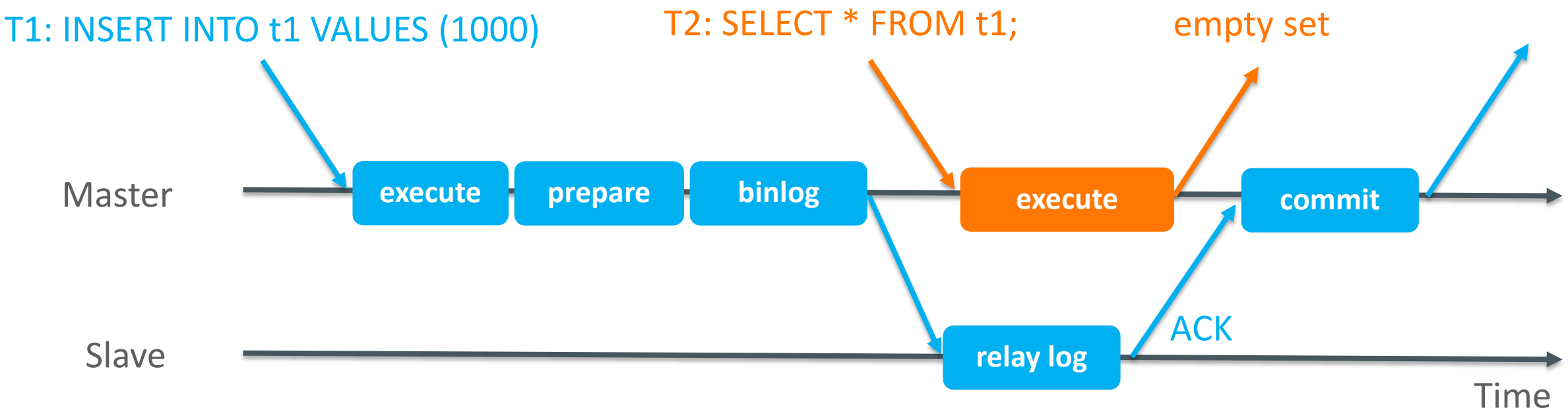
Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Loss-less Semi-sync Replication



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Loss-less Semi-sync Replication



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

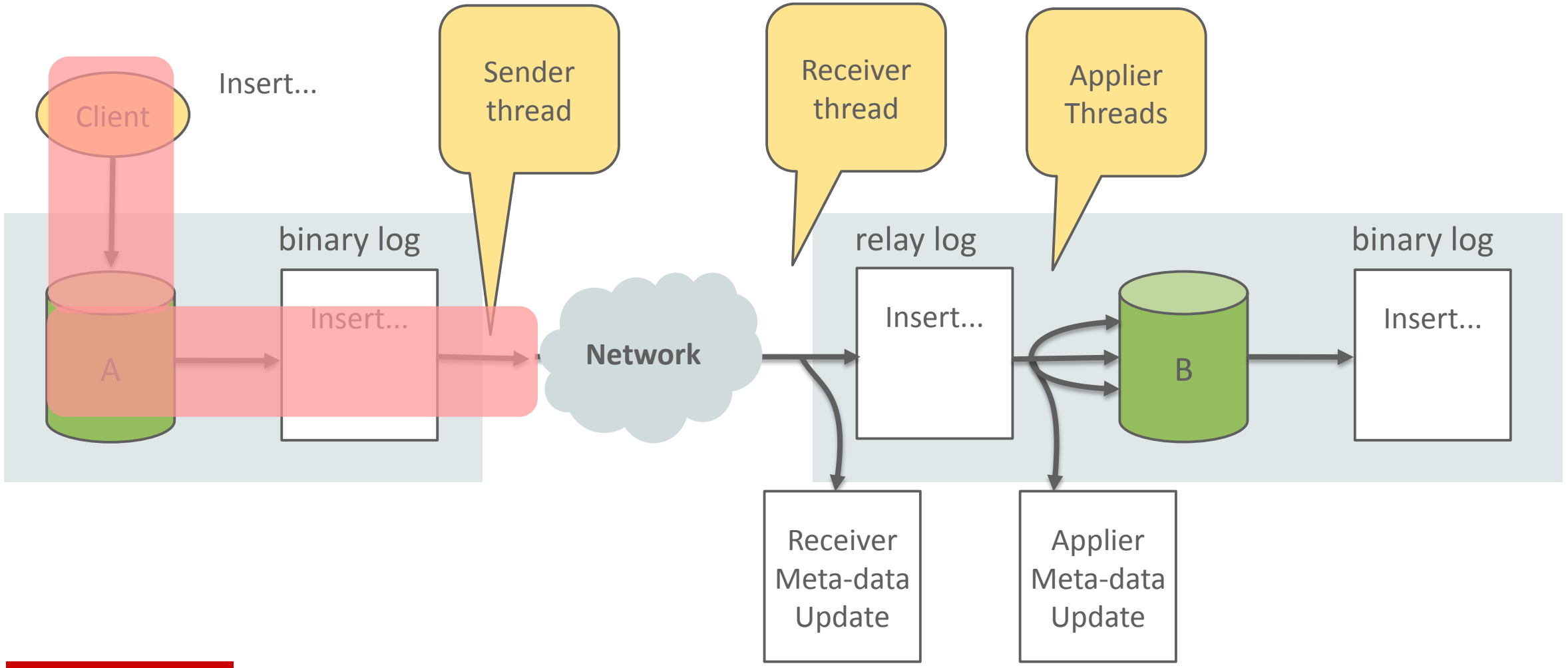
Loss-less Semi-sync Replication

- Master waits for slave's ACK **before** committing (as opposed to: master waits for slave's ACK **after** committing).
  - Therefore, concurrent transactions do not see changes while this transaction waits for ack.
- Should a master fail, then any transaction that it may have externalized is also persisted on a slave.
- User can choose between the original semi-sync behavior and the new

```
mysql> SET rpl_semi_sync_master_wait_point= [AFTER_SYNC|AFTER_COMMIT]
```


Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Semi-sync Replication – Wait for Multiple ACKs



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Semi-sync Replication – Wait for Multiple ACKs

- Master does **not commit** transaction until it receives **N** ACKs from **N** slaves.
- Dynamically settable:

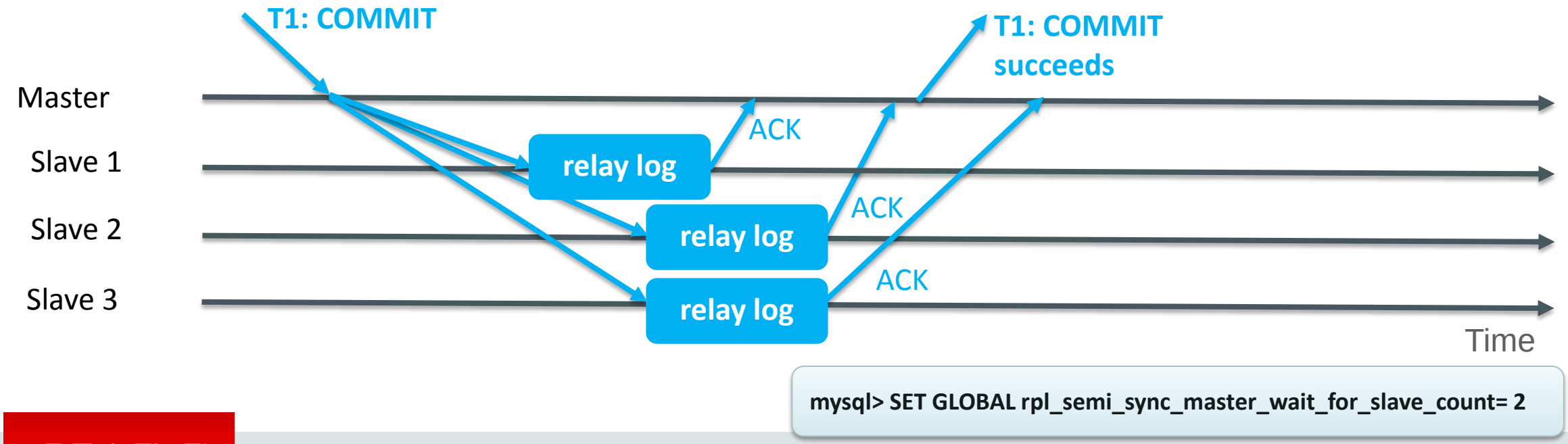
```
mysql> SET GLOBAL rpl_semi_sync_master_wait_for_slave_count= N
```


Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Semi-sync Replication – Wait for Multiple ACKs

- Master does **not commit** transaction until it receives **N** ACKs from **N** slaves.
- Dynamically settable:

```
mysql> SET GLOBAL rpl_semi_sync_master_wait_for_slave_count= N
```



mysql> SET GLOBAL rpl_semi_sync_master_wait_for_slave_count= 2

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

3 The New Replication Features in MySQL 5.7

3.1 Usability / Online

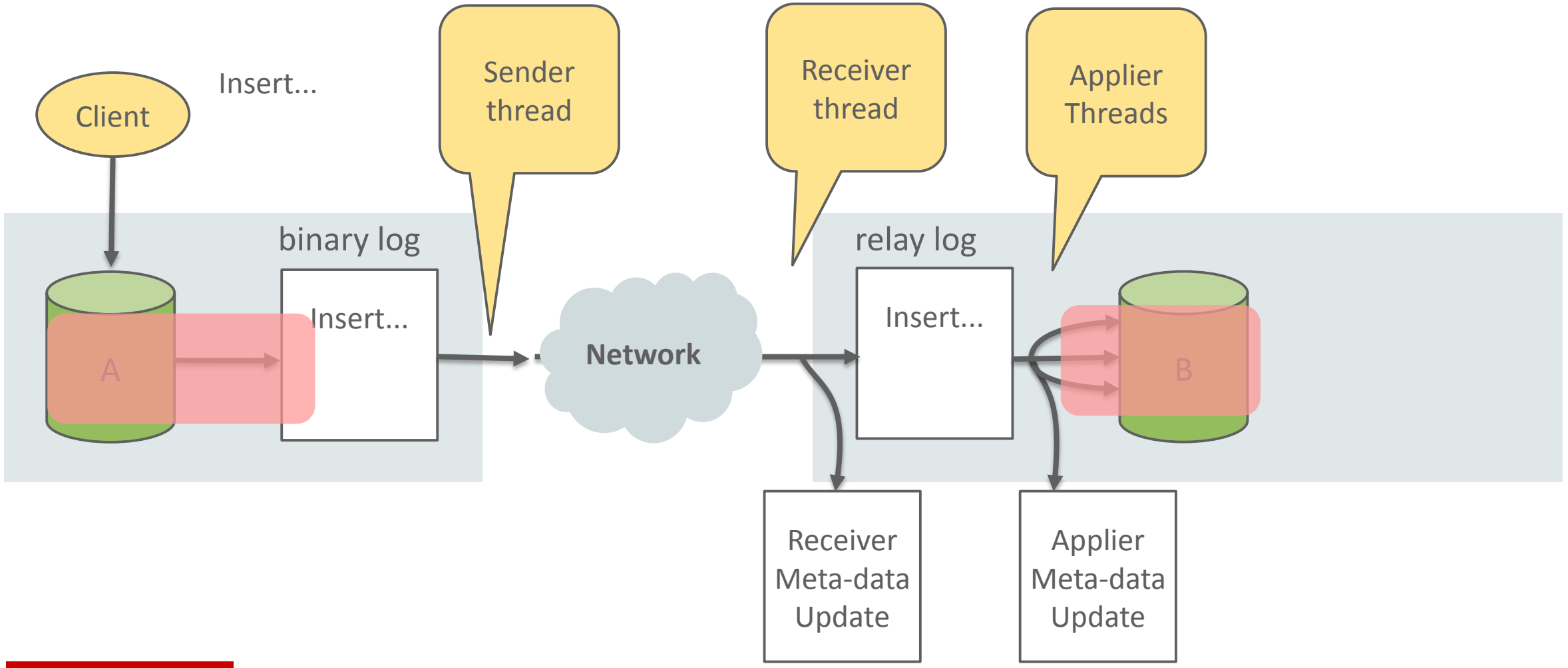
3.2 Performance

3.3 Dependability

3.4 Flexibility

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Storing Global Transaction Identifiers in a Table



Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Storing Global Transaction Identifiers in a Table

Use Cases

- Slaves can use GTIDs with binary logs disabled.
  - Slaves that are never candidates to become a master can still use GTIDs for auto positioning.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Storing Global Transaction Identifiers in a Table

Mechanics

- GTIDs are saved into a mysql system table (range-compressed).

```
CREATE TABLE gtid_executed(
   source_uuid CHAR(36) NOT NULL,
   interval_start BIGINT NOT NULL,
   interval_end BIGINT NOT NULL,
   PRIMARY KEY(source_uuid, interval_start)
 );
```

- GTIDs are **inserted** into the table as transactions commit.
- A compression thread runs periodically and compresses the table into ranges.
  - New global variable controls the period: gtid_executed_compression_period.

```
mysql> SET GLOBAL gtid_executed_compression_period= N;(N – number of transactions)
```


Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

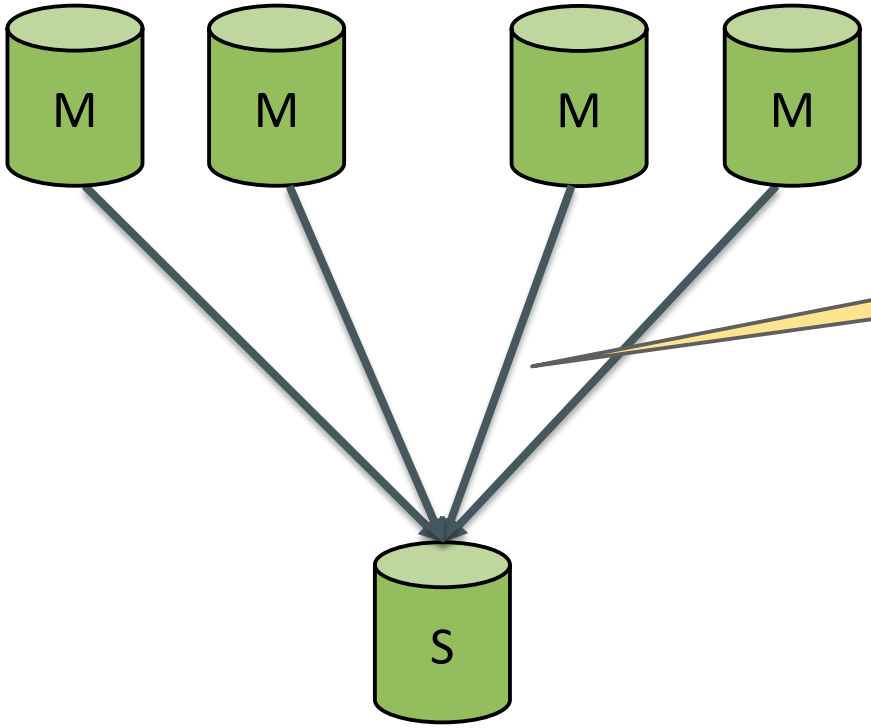
Storing Global Transaction Identifiers in a Table

Mechanics

- Binary Log Enabled
  - Store GTID in binary log (transactionally).
  - Copy GTID set to table on binary log rotation.
- Binary Log Disabled
  - Store GTID in table (transactionally).
  - New transactions do not get an automatic GTID assigned.
- Always store GTIDs transactionally.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Multi-Source Replication



Slave **can** have more than **one master.**

The need for gathering data in a central server:

- Integrated backup;
- Complex queries for analytics purposes;
- Data HUB for inter-cluster replication.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Multi-Source Replication

- A server (slave) can replicate from multiple sources (masters).
- Multiple channels (channel: receiver thread, relay logs, applier threads) that can be stopped started individually.
- Integrated with Multi-threaded Slave (each channel has its own multi-threaded applier set of threads).
- Integrated with the new P_S tables.
  - replication_applier_status_by_coordinator shows multiple entries, one per channel/source coordinator.
  - replication_applier_status_by_worker shows multiple entries, multiple entries per channel
  - replication_connection_status shows multiple entries, one per connection to different sources.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Multi-Source Replication

- Integrated with GTIDs.
- Integrated with crash-safe tables.
  - Progress state is stored in these tables for recoverability purposes.
- Virtually no limit on the number of sources (capped to 256, but can be changed if server binary is rebuilt).
- Able to manage each source separately.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

3 The New Replication Features in MySQL 5.7

- 3.1 Usability / Online
- 3.2 Performance
- 3.3 Dependability
- 3.4 Flexibility
- 3.5 Misc

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

“Smaller”, yet interesting, enhancements!

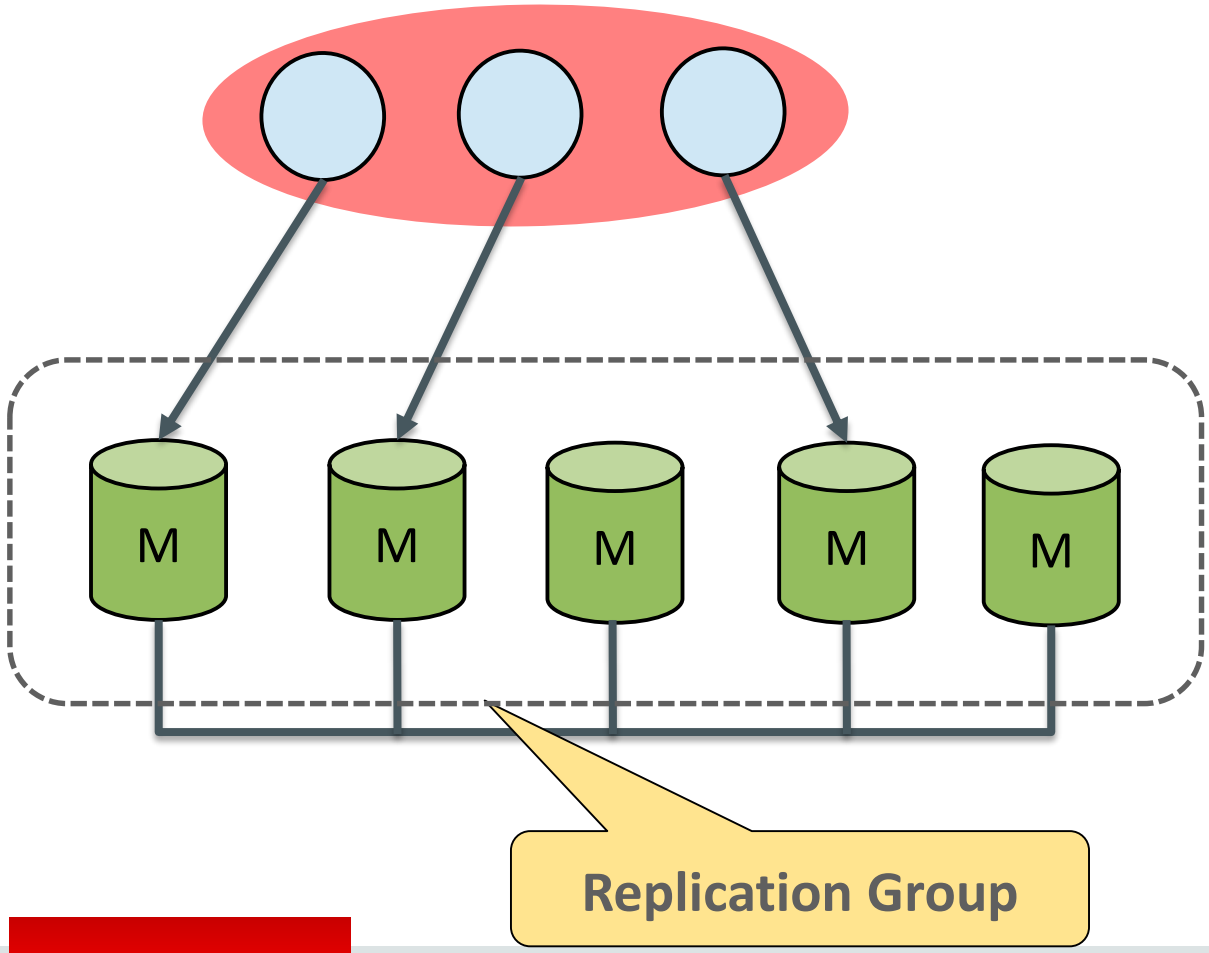
- **Multi-threaded applier** is able to **retry failed transactions**.
- Option to make the **Multi-threaded applier preserve commit order**.
- **SSL** options for **mysqlbinlog** tool.
- **Rewrite DB** rules for the **mysqlbinlog** tool.
- **Function to wait for transactions to be applied**, regardless of the replication stream they come from.
- Options to **track global transaction identifiers** in the **return packet** of the mysql protocol. Useful for tracking GTID session state and enabler for session consistency.
- Support for **XA transactions** when the **binary log is enabled**.
- Change in the defaults. E.g., binlog_format=**ROW** and sync_binlog=**1**.
- Options to **fine tune** the binary log **group commit** procedure.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

4 Development Sneak Peek

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

MySQL Group Replication Plugin



- Multi-master update everywhere
- **Automatic group membership management** and **failure detection**.
- **No need for server fail-over.**
- **Automatic reconfiguration.**
- No single point of failure.
- Shared-nothing state machine replication.
- **InnoDB compliant.** Off-the-shelf hardware friendly.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

5 Roadmap

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

What is Next?

- MySQL **Group Replication**
  - Rapid releases.
  - Improve performance, stability and usability.
- MySQL Replication **Usability**
  - Instrument even more replication and extend replication P_S tables
  - Simpler administrative commands, more online operations
- MySQL Replication **Performance**
  - Continue to improve replication stream multi-threaded (slave) applier
  - Continue to improve replication stream pipeline performance and efficiency

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

6 Summary

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Summary

- Replication feature set in MySQL 5.7 shows many improvements:
  - Semi-sync is more flexible and faster.
  - Less contention on the binary log.
  - Slave is faster and provides an inter-transaction parallelization scheme.
  - More online reconfiguration: Global Transaction Identifiers, filters, configuration.
  - Improved monitoring.
  - Even more flexible replication with multi-source enabling new deployment scenarios.
  - Enhancements all over: transaction retries, sequential commits, XA support, new replication functions, security.

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.

Where to go from here?

- Packages
  - http://dev.mysql.com
  - http://labs.mysql.com
- Reference Documentation
  - http://dev.mysql.com/doc/refman/5.7/en/replication.html
- Blogs from the Engineers (news, quirks, technical information and much more)
  - http://mysqlhighavailability.com

Copyright © 2016, Oracle and/or its affiliates. All rights reserved.


ORACLE®