ORACLE

Inside MySQL 5.7 Replication Features

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

MySQL

Safe Harbour Statement

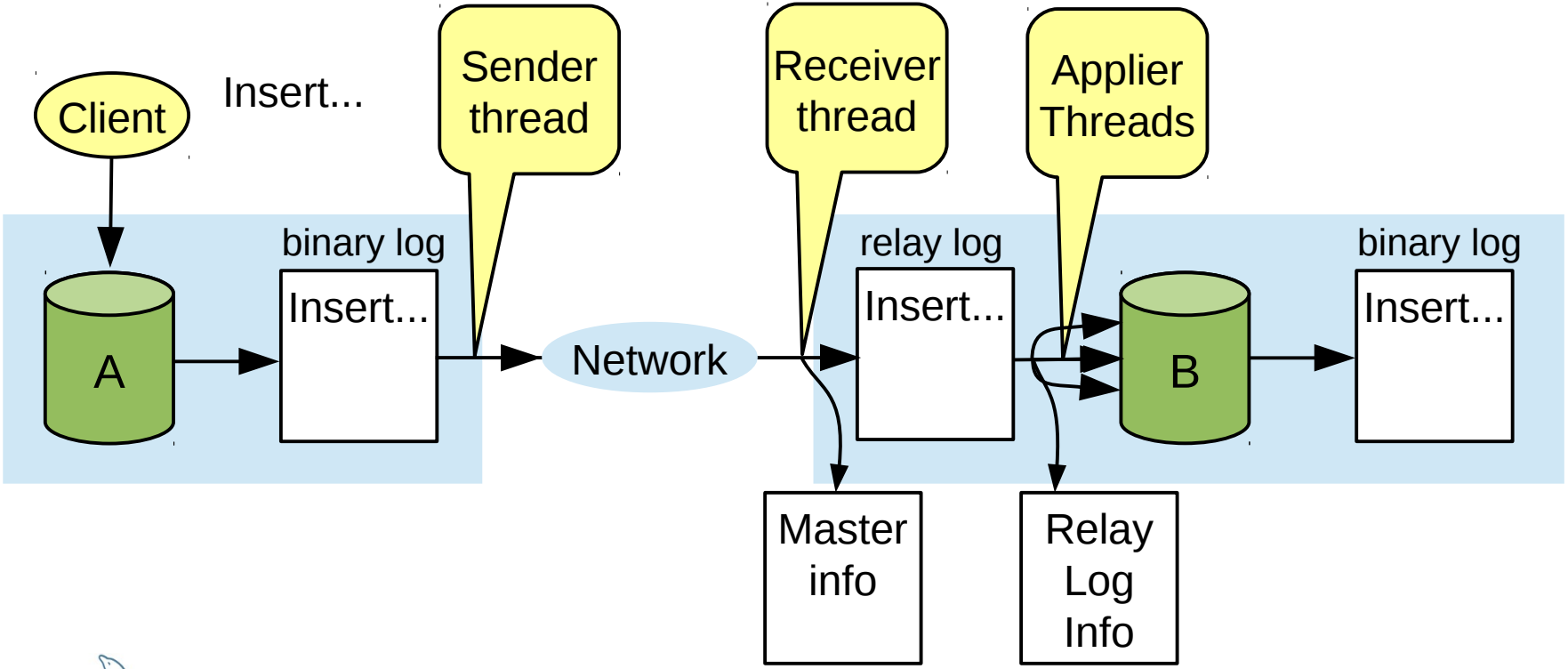
The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract.

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.

Agenda

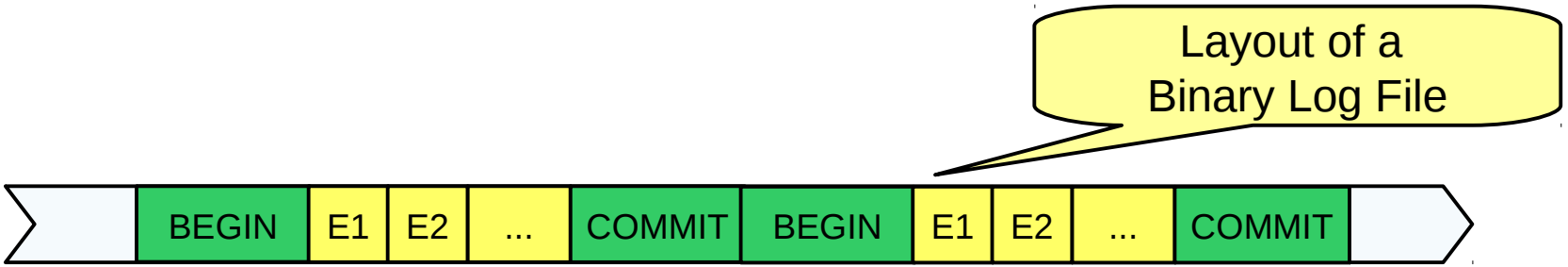
- **Background**
- **Next Generation Features in MySQL 5.7**
- **What is in the Lab?**
- **What is Next?**
- **Summary**

Background: Replication Components



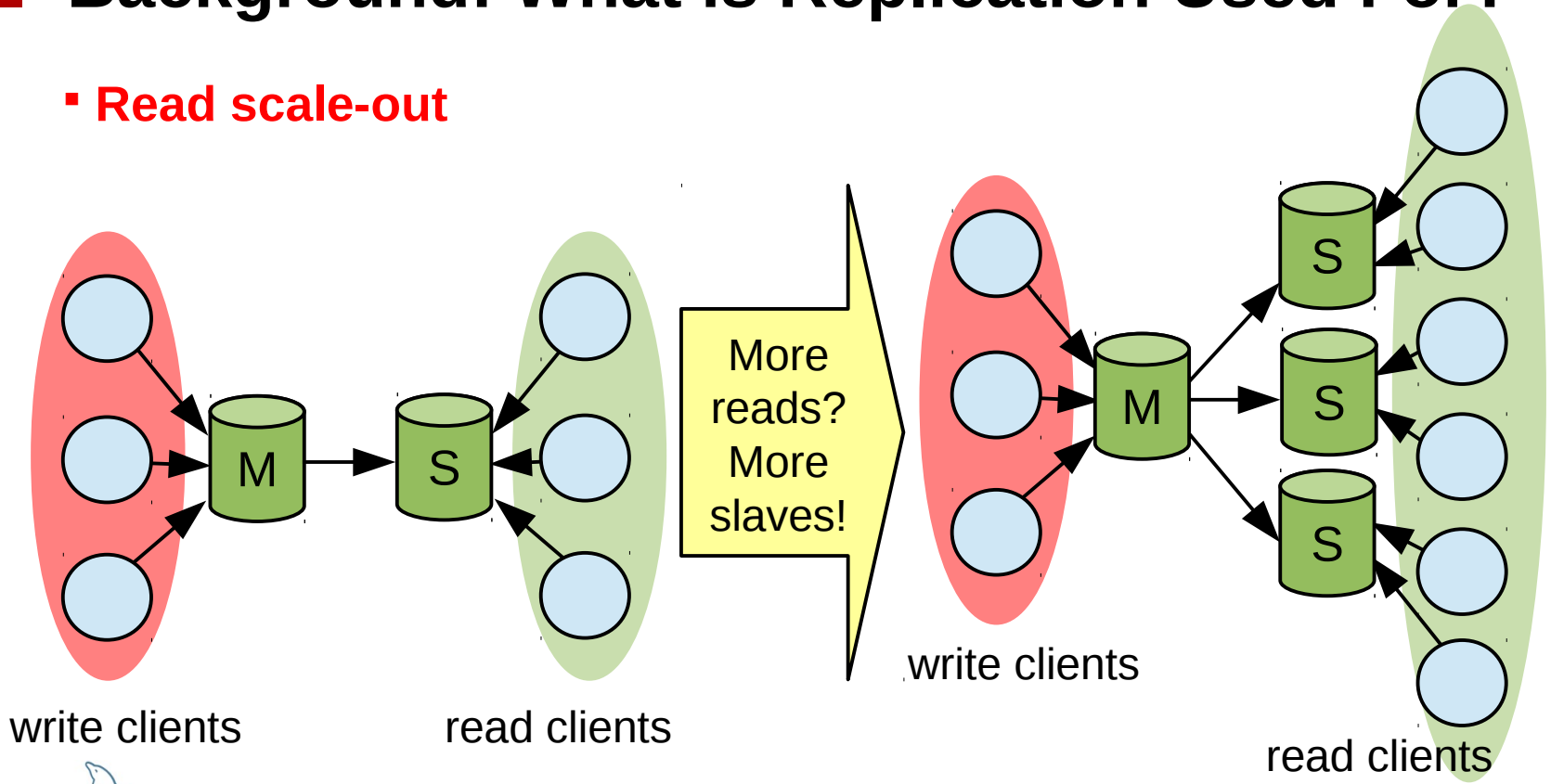
Background: Replication Components

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

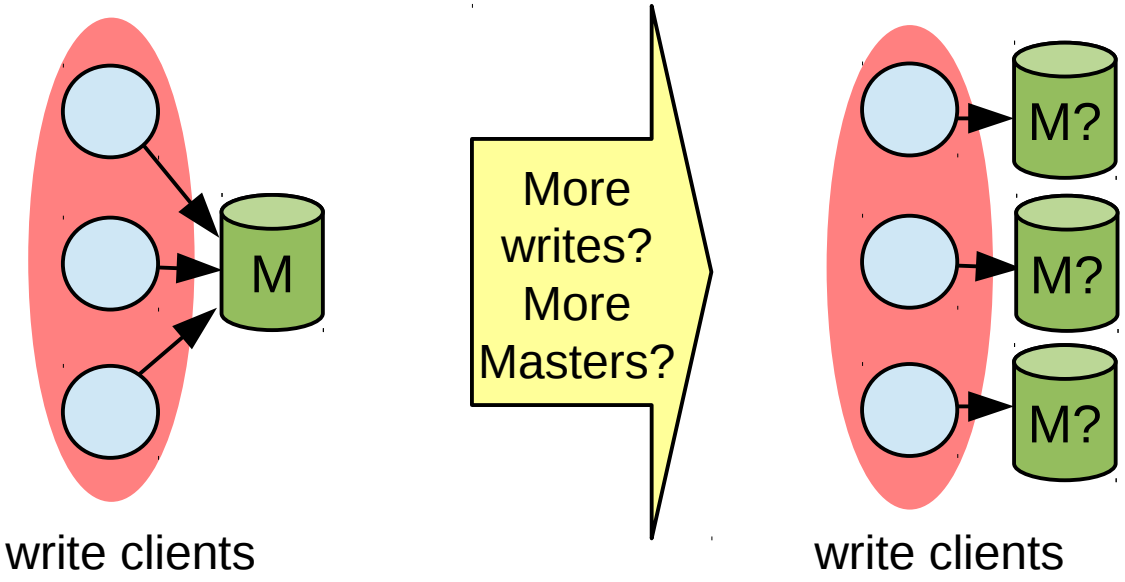


Background: What is Replication Used For?

- **Read scale-out**



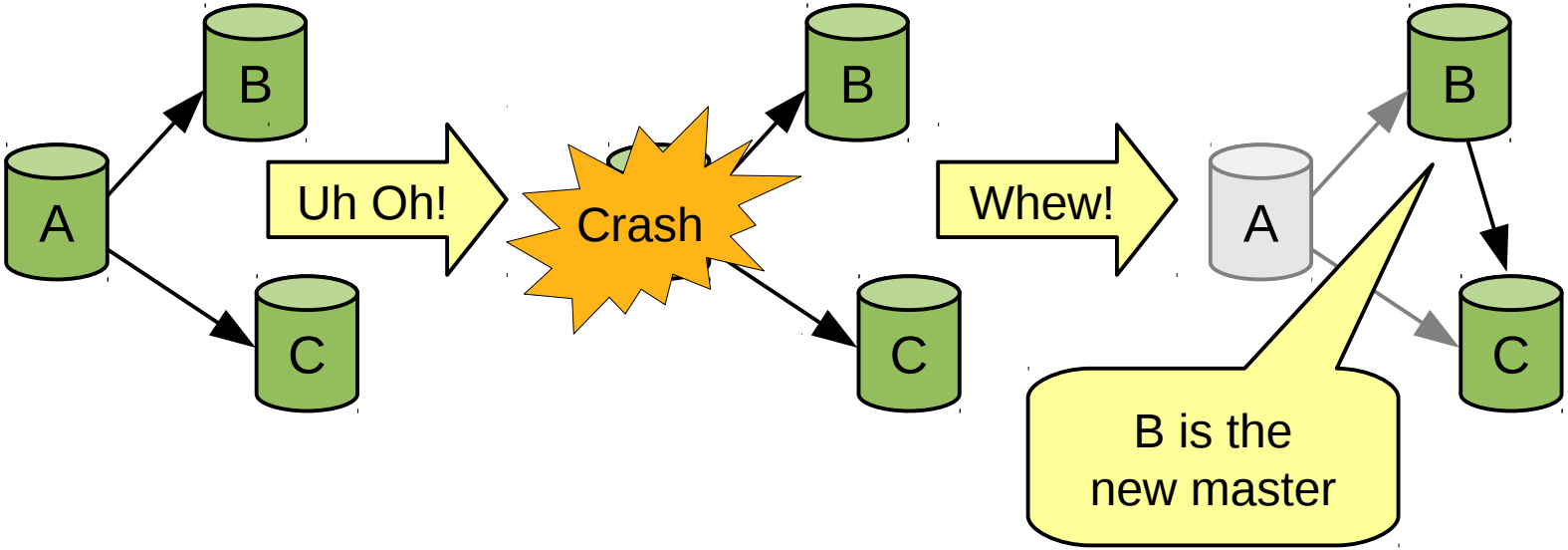
What about Write scale-out?



We will discuss MySQL Fabric later...

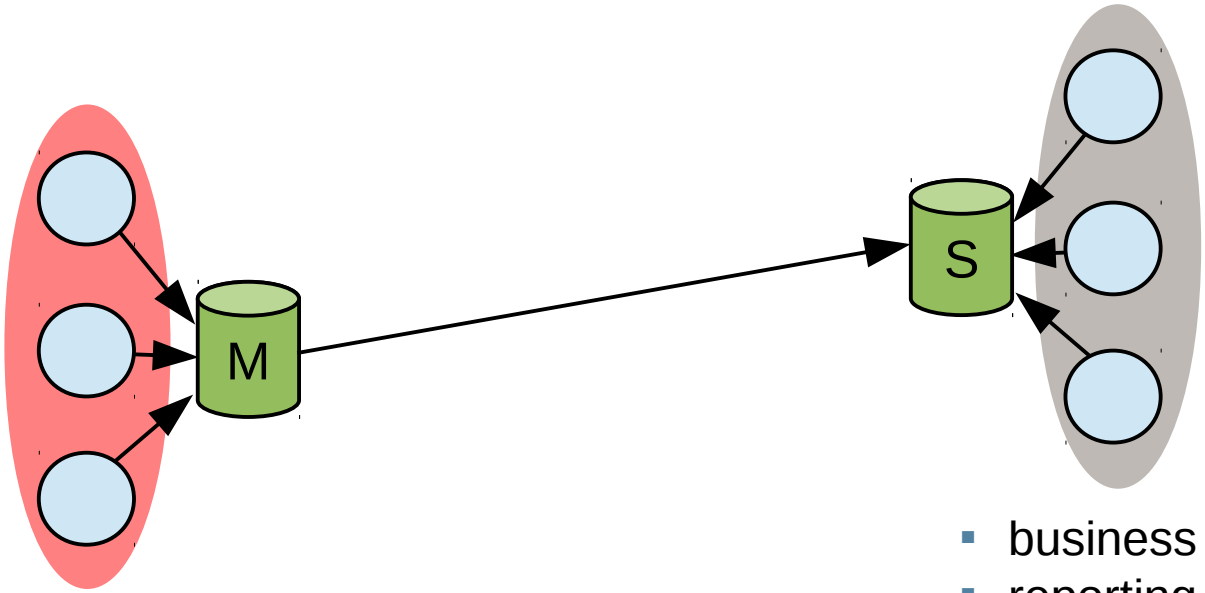
Background: What is Replication Used For?

- **Redundancy**: If master crashes, **promote** slave to master



Background: What is Replication Used For?

- **On-line Backup and Reporting**



- write clients

- business intelligent client apps
- reporting client apps
- big queries client apps

Background: What is Replication Used For?

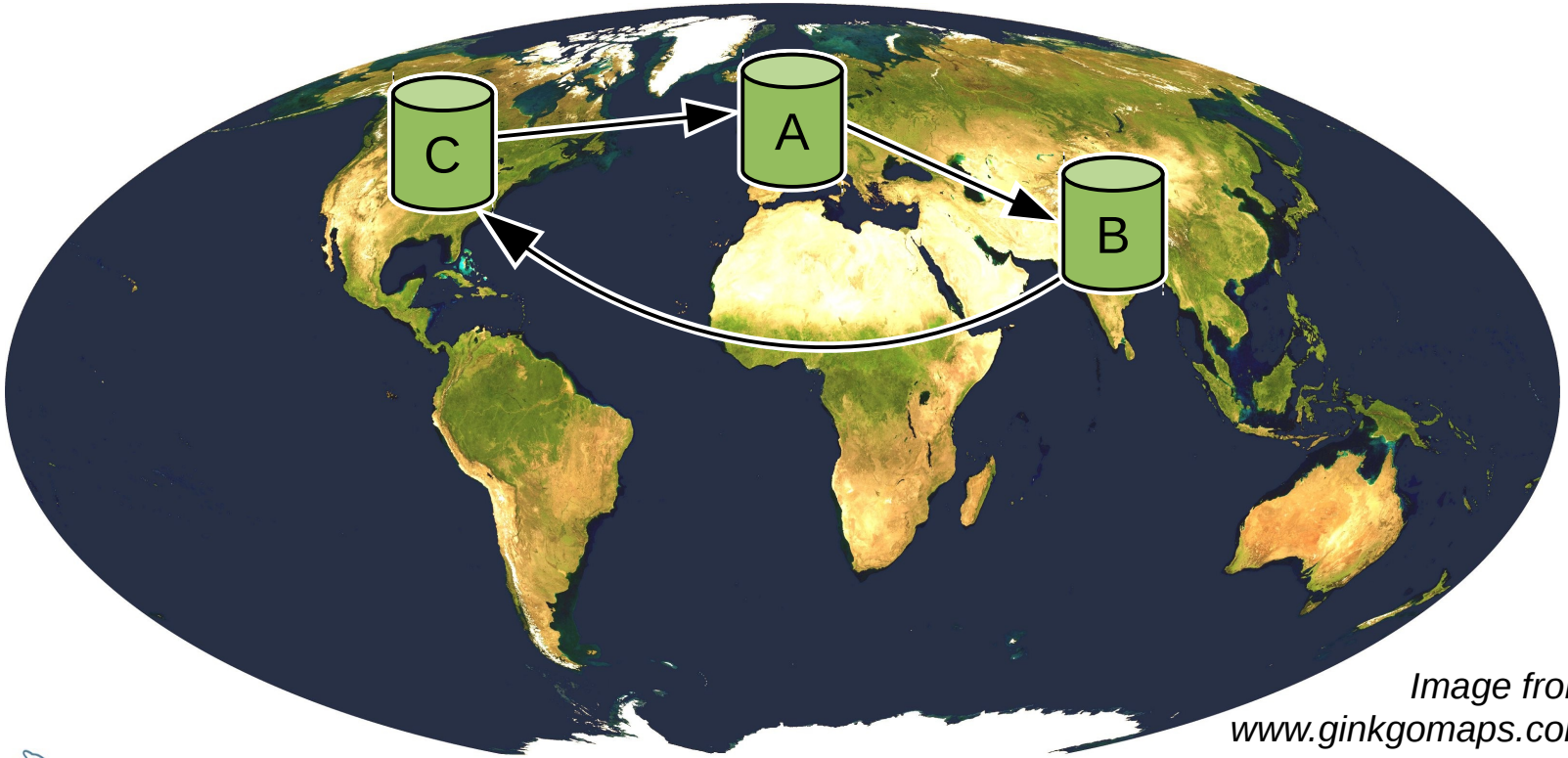


Image from www.ginkgomaps.com

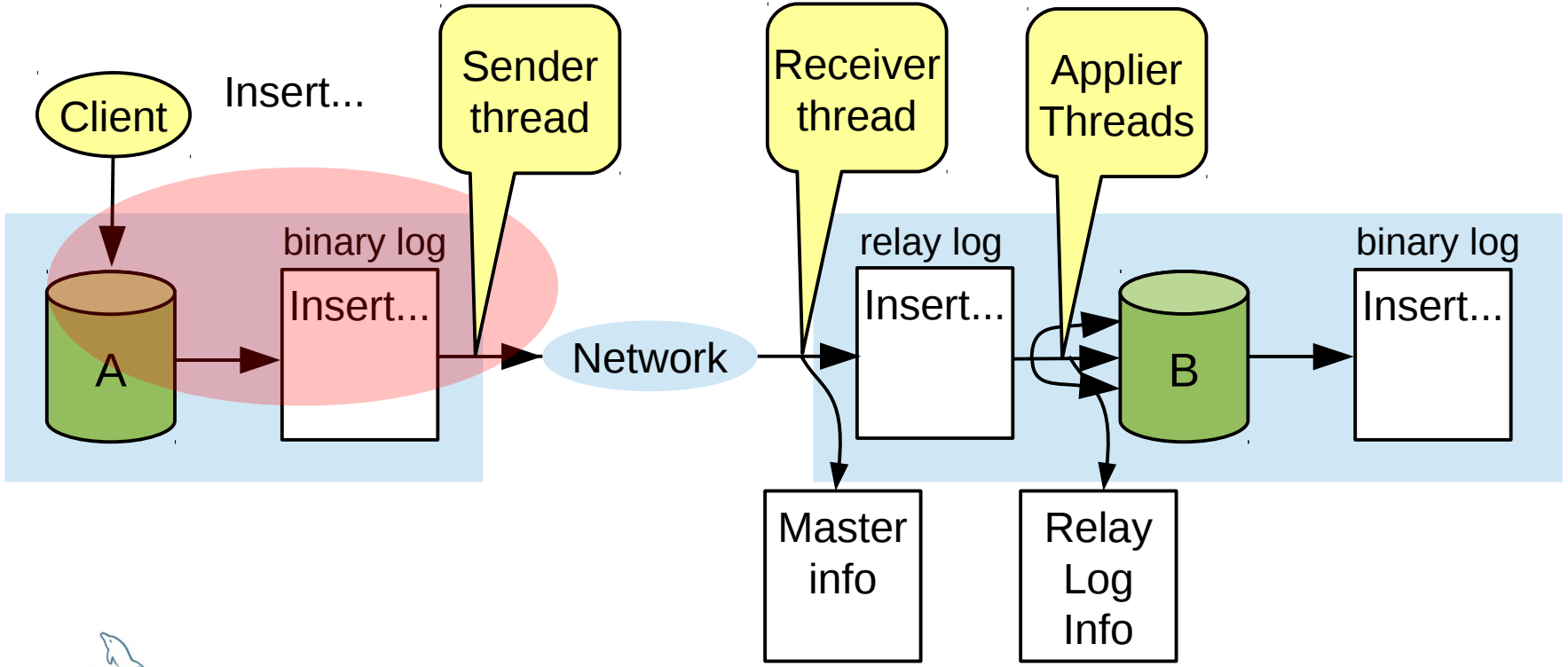
Agenda

- Background
- **Next Generation Features in MySQL 5.7**
- What is in the Lab?
- What is Next?
- Summary

What is New? (2013 –)

- **MySQL 5.7.2** - **Development Milestone Release, September 2013**
  - Higher Master Throughput
    - Sender, aka Dump, Thread Does Not Take the Binary Log Lock.
  - Higher Slave Throughout
    - Multi-Threaded (Slave) Timestamp based Applier (MTS).
  - Better Monitoring of Replication
    - Instrumentation for getting replication status through performance schema.
  - Loss-less Semi-sync Replication.

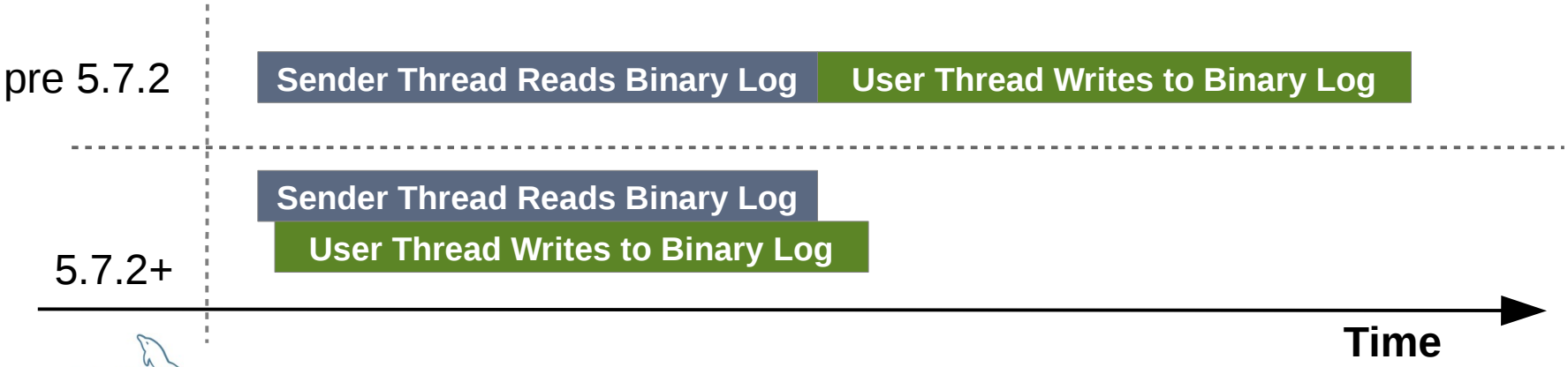
Higher Master Throughput: Sender, aka Dump, Thread Enhancement



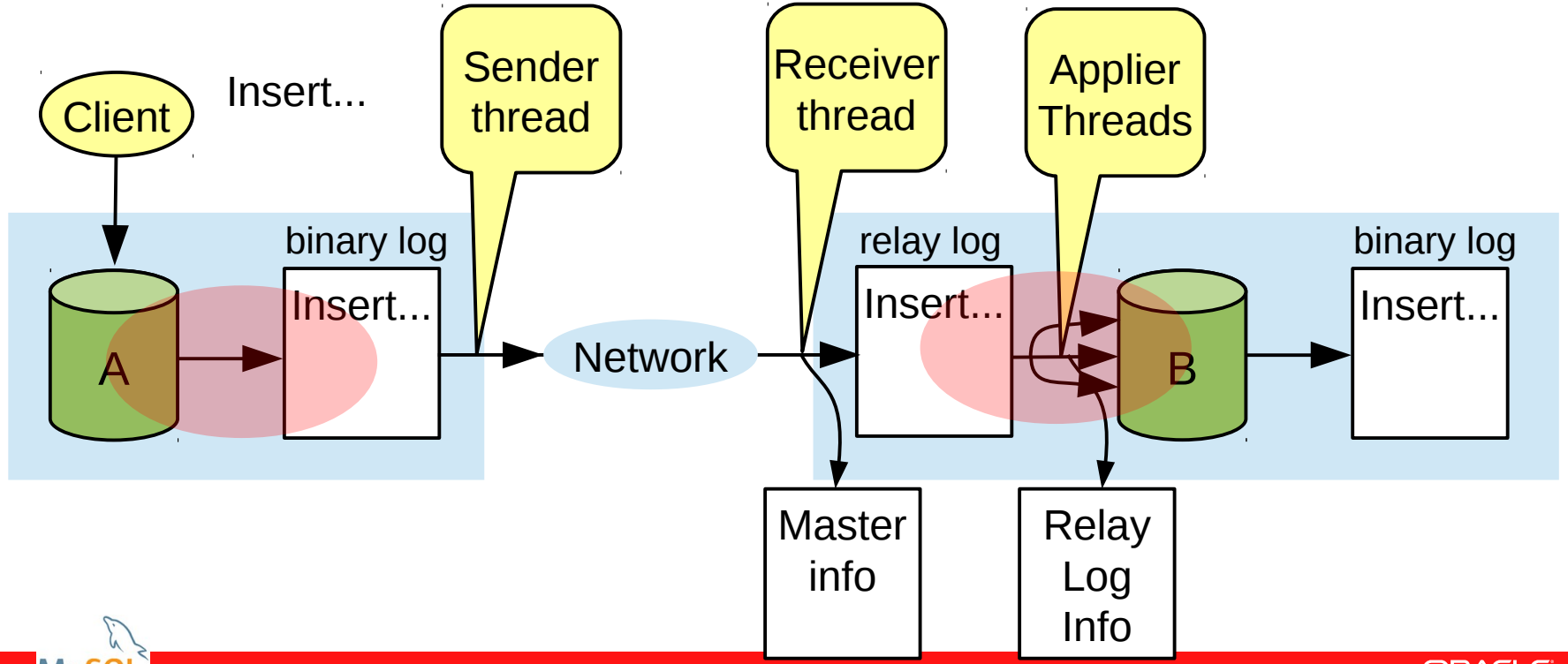
Higher Master Throughput: Sender, aka Dump, Thread Enhancement

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.**



Higher Slave Throughput: Timestamp based Multi-threaded Applier

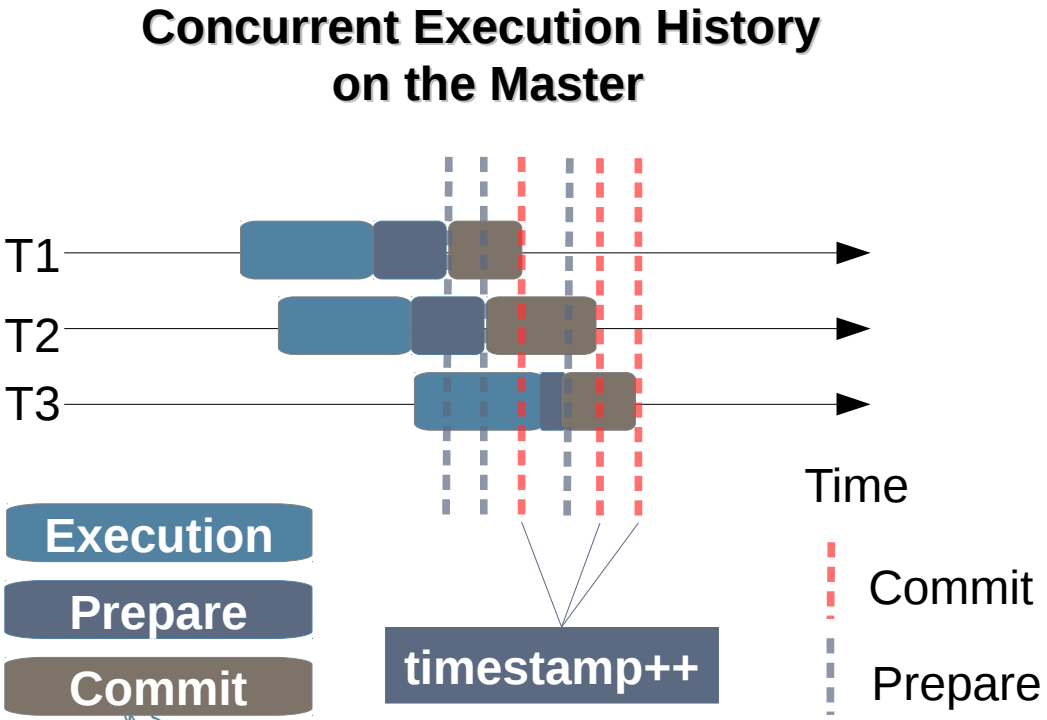


Higher Slave Throughput: Timestamp based Multi-threaded Applier

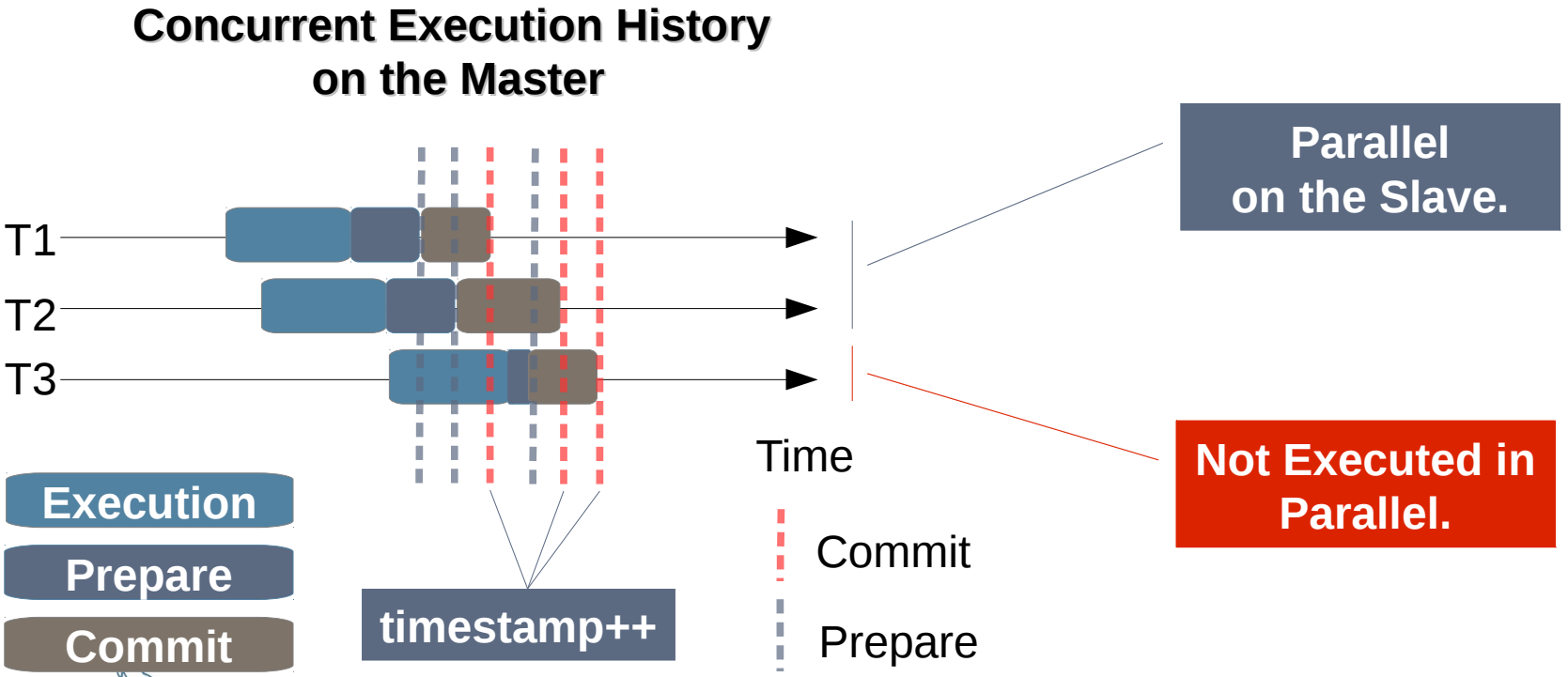
- Leverage parallelization information obtained from the execution on the master.
  - Transactions that prepare on the same "*version*" of the database, are assigned the same timestamp.

- Meanwhile, at the slave:
  - Transactions with the same timestamp can be executed in parallel;
  - Concurrent transactions commit independently, thus no waiting involved.

Higher Slave Throughput: Timestamp based Multi-threaded Applier



Higher Slave Throughput: Timestamp based Multi-threaded Applier



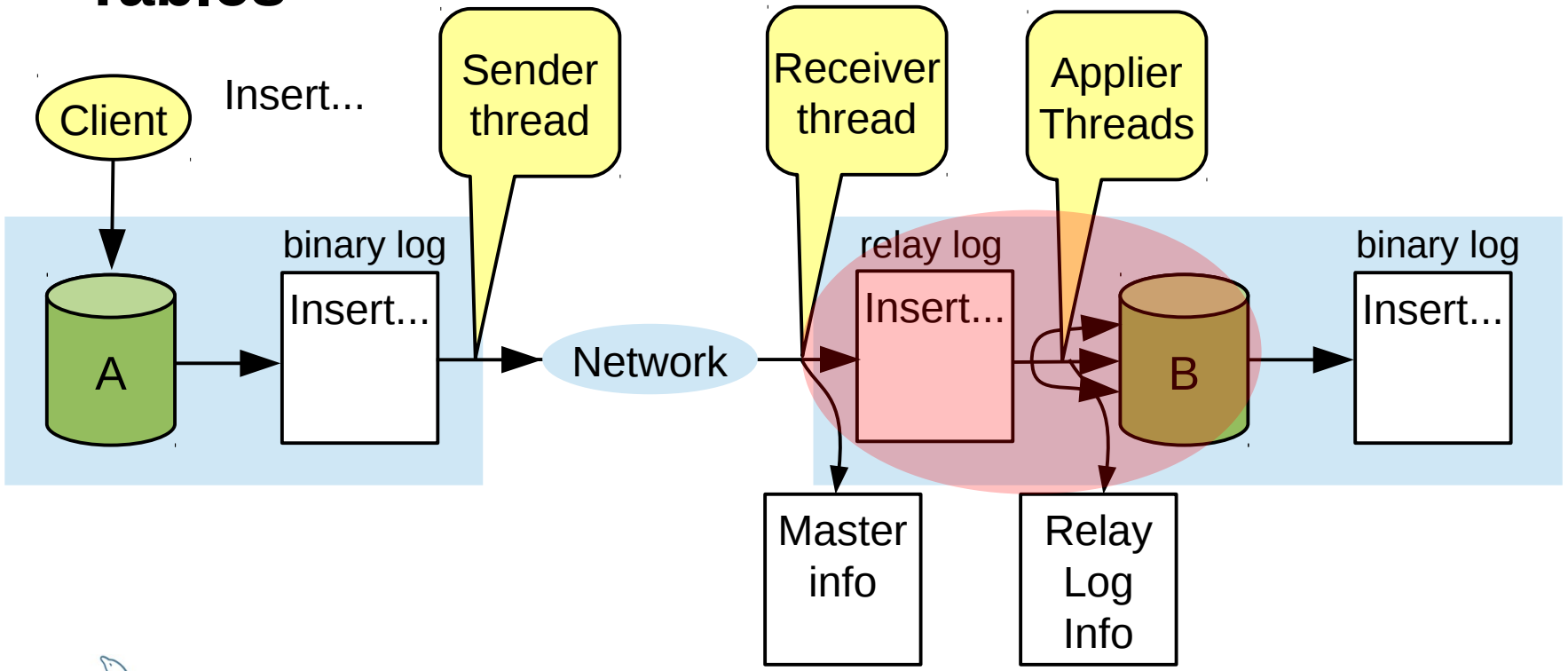
Higher Slave Throughput: Timestamp based Multi-threaded Applier

- 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 prepare timestamp
  - *database* - the scheduling policy from 5.6 (concurrency control done per database).
- Work to improve slave scalability continues, does not stop here.

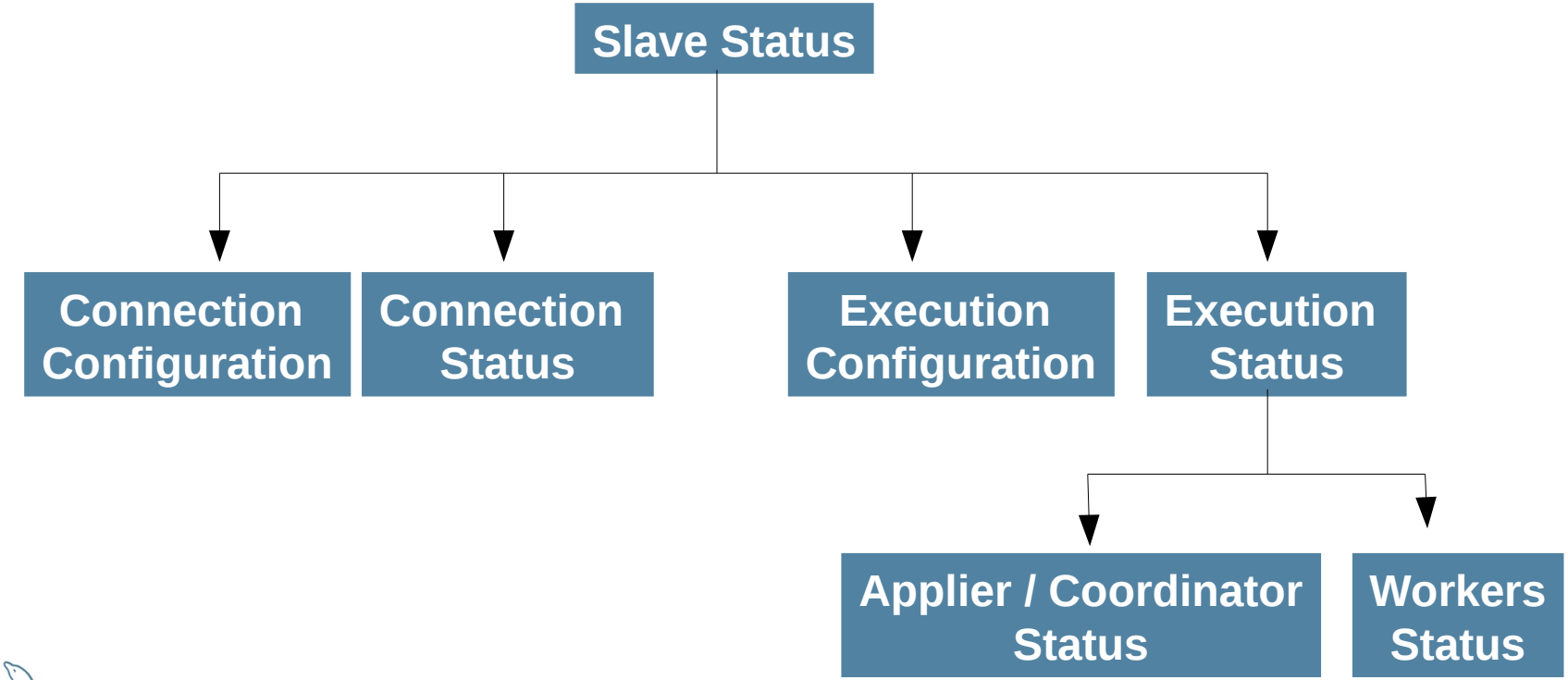
Better Replication Monitoring: P_S Replication Tables



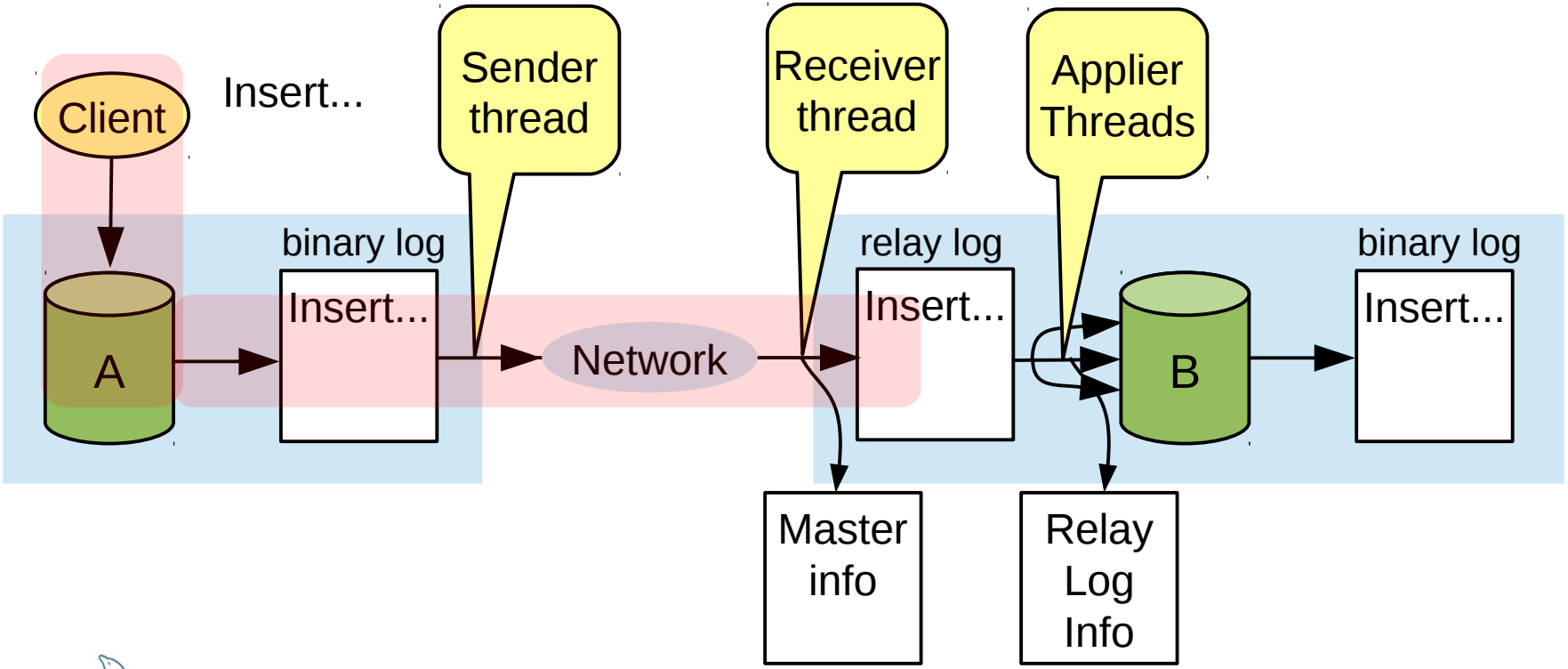
Better Replication Monitoring: P_S Replication Tables

- Access monitoring information through an SQL interface.
- Write stored functions or procedures with input from replication internals.
- Logically unrelated information into different places.
- Flexible and easier to extend and adapt as new feature get into the server.
- More user friendly names identifying the monitoring fields.

Better Replication Monitoring: P_S Replication Tables



Loss-less Semi-sync Replication

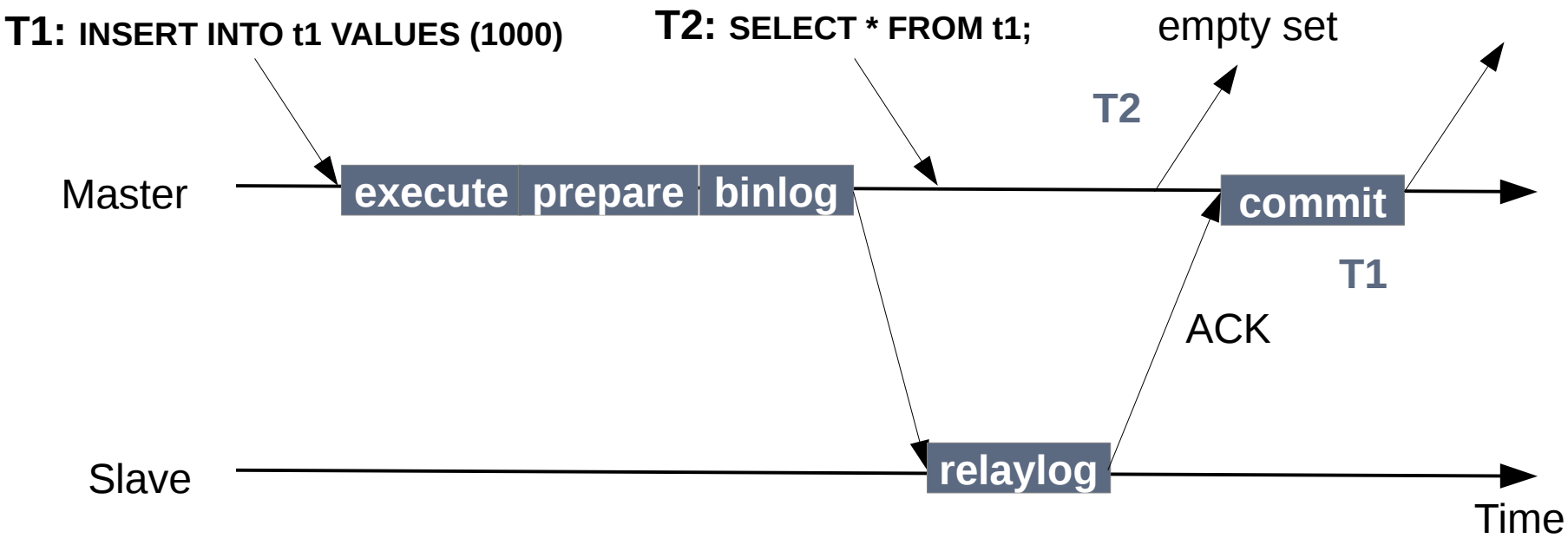


Loss-less Semi-sync Replication

- Master does not **commit** transaction until it receives and ACK from the slave.
  - (as opposed to: Master does not **release the session** until it receives and ACK from the slave.)
  - Therefore, concurrent transactions do not externalize changes while waiting 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 semisync behavior and the new one.

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

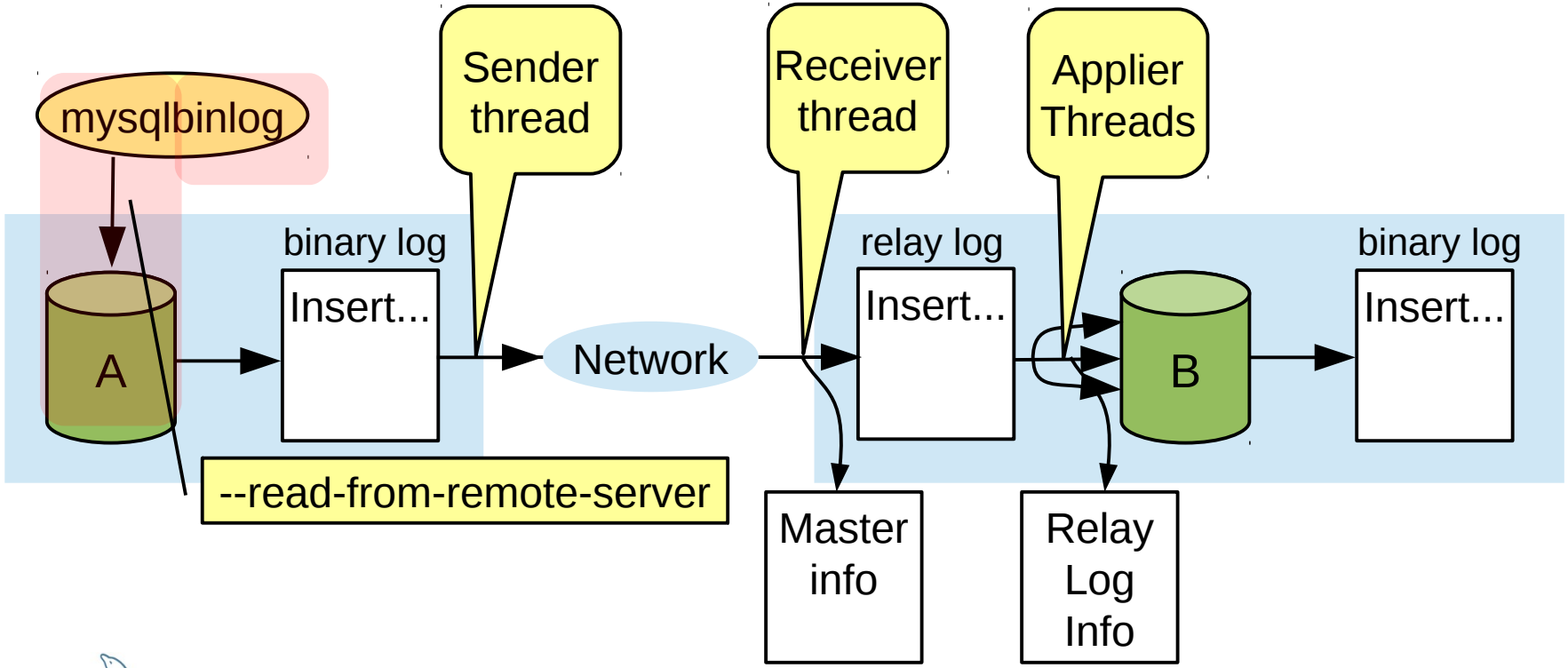
Loss-less Semi-sync Replication



What is New? (2013 –)

- **MySQL 5.7.3** - **Development Milestone Release, December 2013**
  - Improved Security
    - SSL options for mysqlbinlog
  - Flexible Semisync Durability
    - Configure master to wait for more than one semisync slave to ACK back.
  - More Production Friendly
    - Changing Replication Filters Dynamically.

SSL options for mysqlbinlog



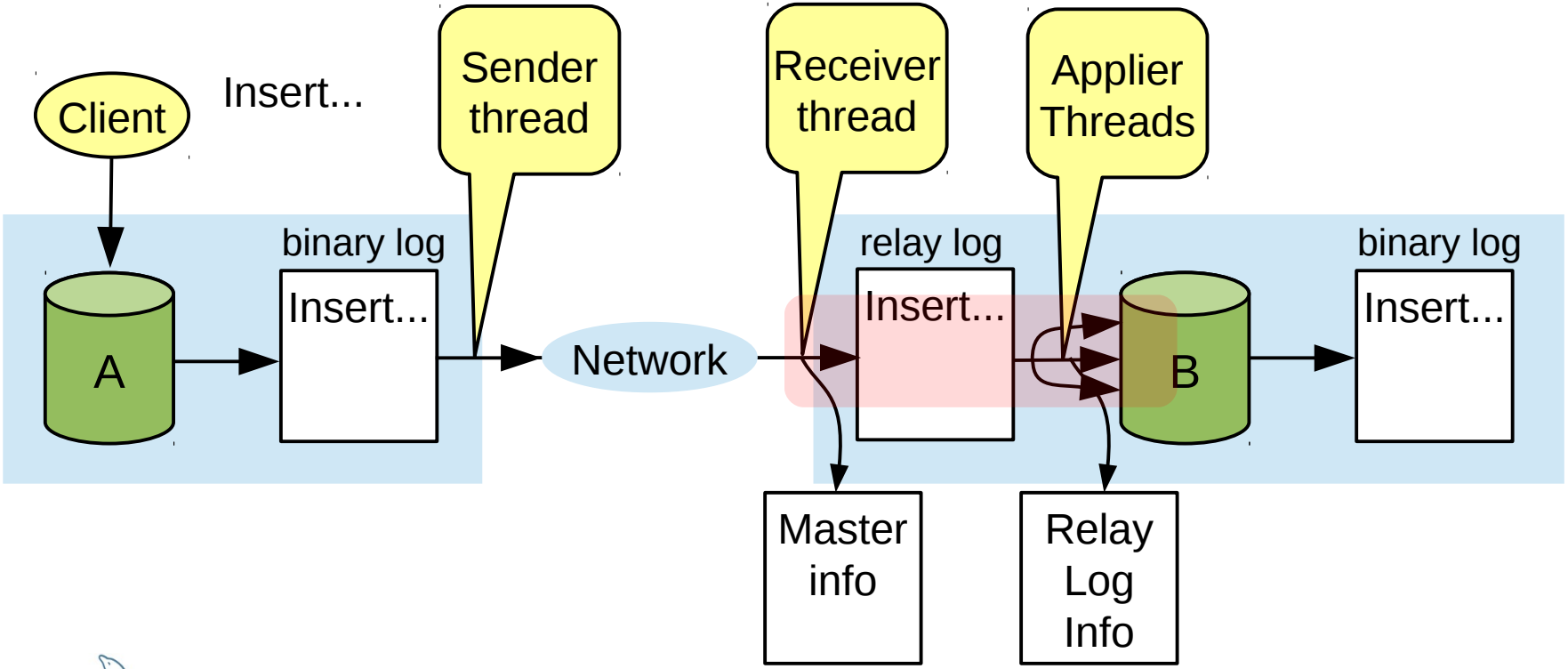
SSL options for mysqlbinlog

- --ssl* options to mysqlbinlog
  - Reading binary logs from remote servers through a secure channel.
  - Supports all SSL options that other client tools support.

```
mysql>  GRANT USAGE ON *.* TO 'rpluser'@'localhost' REQUIRE SSL;
```

```
shell>  mysqlbinlog --read-from-remote-server -ssl -u rpluser ...
```

Dynamic Slave Replication Filters

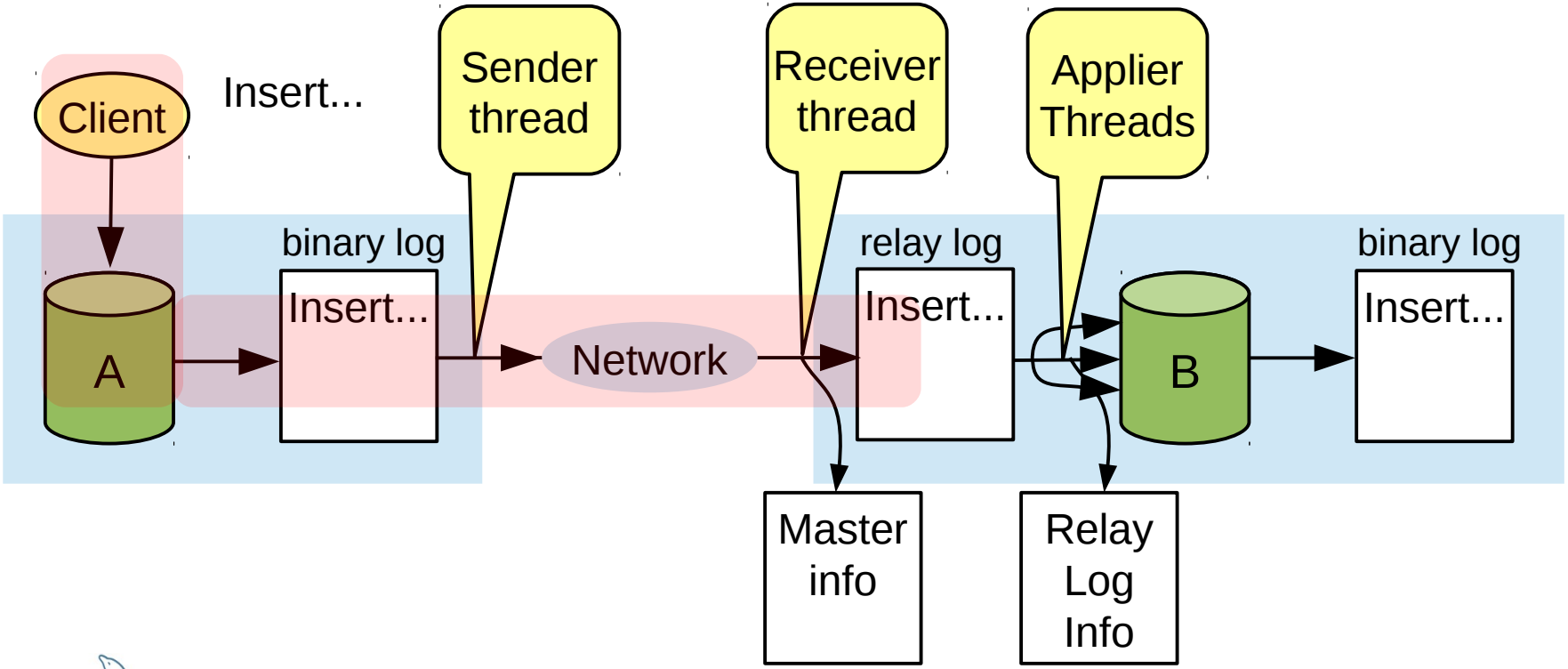


Dynamic Slave Replication Filters

- Change Slave's Replication Filters dynamically.
  - No need to stop and restart slave for establishing 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)
```

Semi-sync Replication – Wait for Multiple ACKs



Semi-sync Replication – Wait for Multiple ACKs

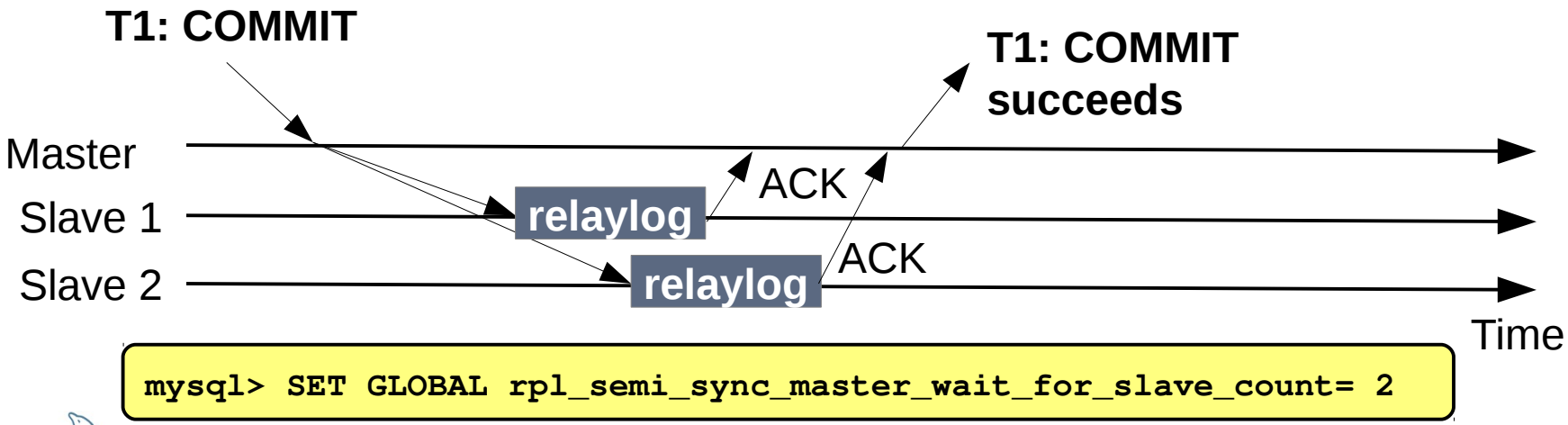
- Master does not **commit** transaction until it receives **N** ACKs from the slave.
- Dynamically settable:

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

Semi-sync Replication – Wait for Multiple ACKs

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

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



Agenda

- Background
- Generally Available Features in MySQL 5.7
- Next Generation Features in MySQL 5.7
- **What is in the Lab?**
- What is Next?
- Summary

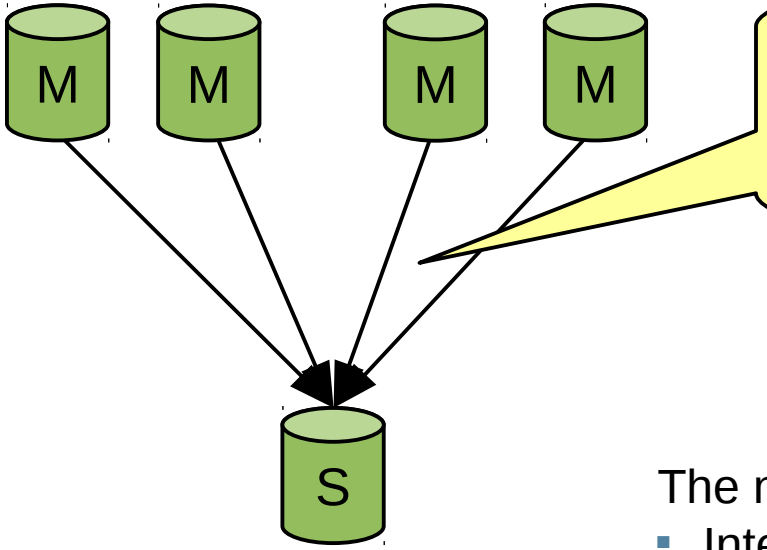
Multi-Source Replication

- A server (slave) can replicate from multiple sources (masters).
- Multiple channels (channel: connection thread, relay log, 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_execute_status_by_coordinator shows multiple entries, one per channel/source applier.
  - replication_connection_status shows multiple entries, one per connection to different sources.

Multi-Source Replication

- Integrated with GTIDs.
- Integrated with crash-safe tables.
  - Progress state is stored in these tables for recoverability purposes.
- Works with semisync replication.
  - Working on further improving the integration.
- No limit on the number of sources.
- Able to manage each source separately.

Multi-Source Replication



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

Feature preview based on **5.7** is available on **labs.mysql.com .**

The need for gathering data in a central server:

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

Write Scale-Out - Fabric

- Big workloads, Replication helps by offloading the master w.r.t. read queries (which are redirected to slaves).
- Huge workloads, Replication helps but eventually the requirement to write everywhere sets a limit on the scalability.
- **SOLUTION:** Split the database into chunks (shards) and distribute writes for groups of servers, instead of writing everywhere.
- **CHALLENGES:** routing queries, global tables, shard-aware connectors, shard migration, shart splitting, cross-shard joins, ...
- Enter **MySQL Fabric.** (Dr. Lars Thalmann will speak about that later today.)

Agenda

- Background
- Generally Available Features in MySQL 5.7
- Next Generation Features in MySQL 5.7
- What is in the Lab?
- **What is Next?**
- Summary

What is Next?

- MySQL Replication **Usability**
  - Instrument even more replication and extend replication P_S tables
  - Simpler administrative commands
  - Continue to build the GTIDs infrastructure
- MySQL Replication **Availability**
  - Continue to improve semi-sync
- MySQL Replication **Performance**
  - Continue to improve slave performance and improve multi-threaded slave
- MySQL **Fabric**
  - Extend High-Availability Support
  - Work with community on development

Our focus areas

- **Read Scale-out**: Increased performance
  - MySQL Replication
- **Usability**: Easier to use
  - MySQL Utilities
- **Loss-less replication**: Never loose information
  - MySQL Semi-synchronous Replication
- **Write Scale-out**: Increased performance
  - MySQL Fabric
- **High-Availability**: System always accessible
  - MySQL Fabric

Work on improvements

Work on improvements

Work to get to GA

Work on more features

Agenda

- Background
- Generally Available Features in MySQL 5.7
- Next Generation Features in MySQL 5.7
- What is in the Lab?
- What is Next?
- **Summary**

Summary

- MySQL 5.7.2 already shows features improving:
  - Performance, Availability and Usability.
- MySQL 5.7.3 continued down that path:
  - Tools got improved, HA enhancements, usability improvements.
- Lab releases provide a sneak peek at what is cooking: Multi-source, Fabric.

Next Steps: Read about the 5.7 DMRs Replication Features

- Read more about **P_S tables for replication** on Shiv's blog:

  http://shivjijha.blogspot.com/2013/09/Monitoring-Replication-with-the-NEW-performance-schema-tables.html

- Read more about **Intra-Schema Multi-threaded Slave** on Rohit's blog:

  http://geek.rohitkalhans.com/2013/09/enhancedMTS-deepdive.html

- Read more about **Semisync enhancements** on Libing's blog:

  http://my-replication-life.blogspot.com/2013/09/dump-thread-enhancement.html

  http://my-replication-life.blogspot.pt/2013/09/loss-less-semi-synchronous-replication.html

Next Steps: Read about the 5.7 DMRs Replication Features

- Read more about **mysqlbinlog idempotent mode** on Rohit's blog:

  http://binlogtorelaylog.blogspot.com/2013/05/mysqlbinlog-idemmpotent-mode.html

- Read more about **mysqlbinlog –rewrite-db option** on Manish's blog:

  http://manishthe.blogspot.com/2013/05/introduction-with-wonderful-and-best.html

Next Steps: Read about the 5.7 DMRs Replication Features

- Read more on **Dynamic Replication Filters** on Venkat's blog:

  http://my-s-q-l.blogspot.com/2013/12/Making-MySQL-Slave-Replication-Filters-Dynamic.html

- Read more on **Semi-sync Multiple ACKs** on Libing's blog:

  http://my-replication-life.blogspot.com/2013/12/enforced-semi-synchronous-replication.html

- Read more on **SSL options for mysqlbinlog** on João's blog:

  http://jmysqlrep.blogspot.com/2013/12/mysql-57-mysqlbinlog-now-supports-ssl.html

Next Steps: Read about the Replication Features on labs.mysql.com

- Read more about **Multi-source replication** on Rith's blog:

  http://on-mysql-replication.blogspot.com/2013/09/feature-preview-mysql-multi-source-replication.html

QUESTIONS?


ORACLE®