
ORACLE®

ADVANCED MYSQL REPLICATION ARCHITECTURES

Luís Soares
Sr. Software Developer
MySQL Replication Team Lead

Lars Thalmann, Director Replication, Backup, Utilities and Connectors
Mats Kindahl, Lead Software Dev, Replication

OSCON, July, 2012

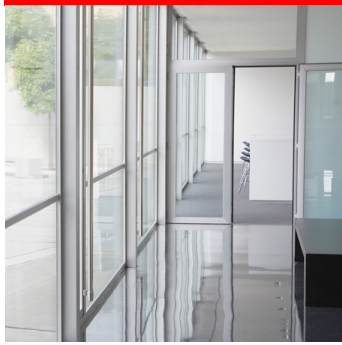
Safe Harbor Statement

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 © 2012, Oracle and/or its affiliates. All rights reserved

Agenda



- MySQL Replication Basics
- Traditional Architectures
- Load Balancing
- Data Aggregation and Multi-source Replication
- Hierarchical Replication
- Data Integration
- MySQL 5.6 and Advanced Replication Architectures Enablers

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

MySQL Replication Basics

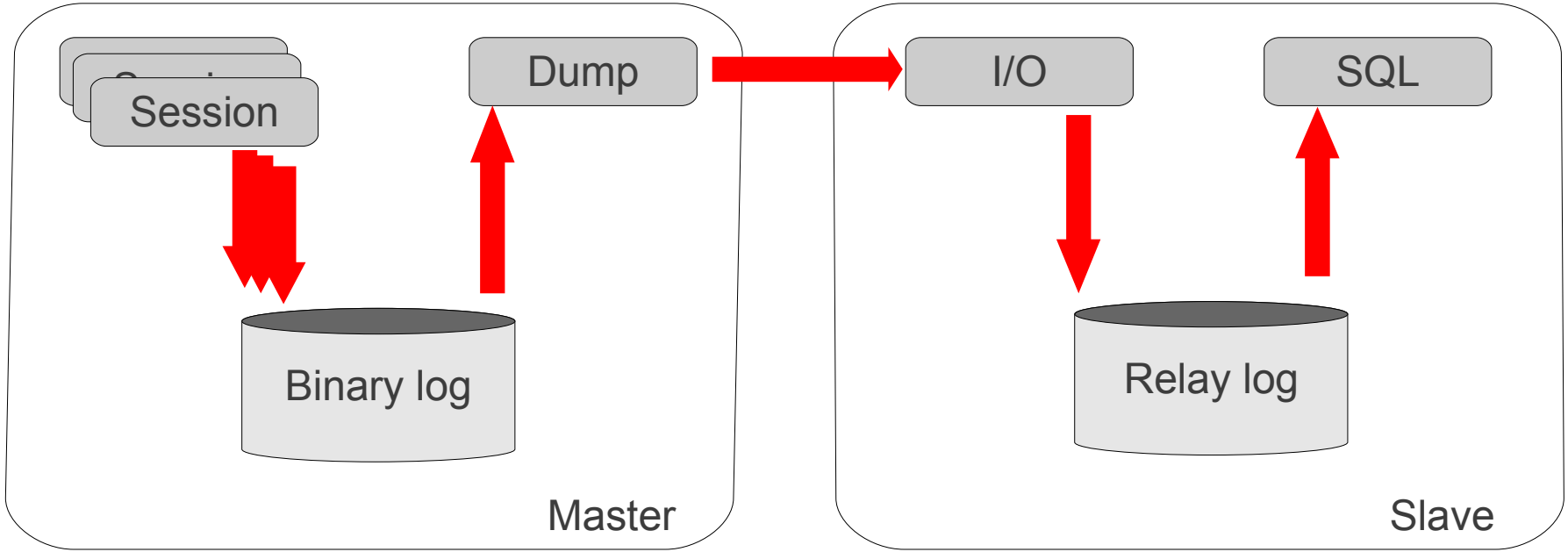
Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Components

- Master
  - Changes data
  - Logs changes (events) into a file (the binary log)
- Slave
  - Retrieves events from the master
  - Replays the events on the slaves databases
- Binary Log
  - Records every change on the master (in either format: *row* or *statement*)
  - Split into transactional groups

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Big Picture



Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Propagating the Changes

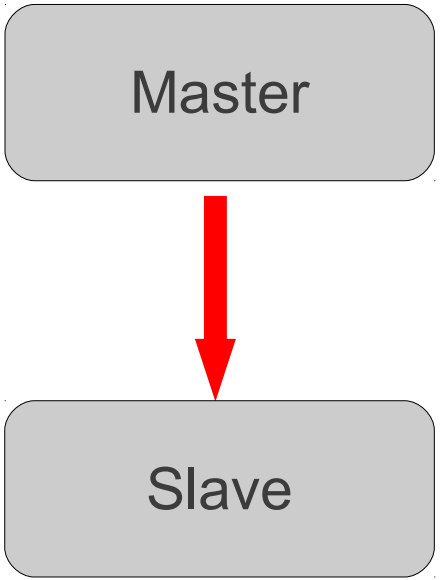
- Asynchronous Replication
  - Transactions are committed and externalized without interaction with the replication layer.
  - Events are propagated after the commit operation is acknowledged.
  - Faster but vulnerable to lost updates on server crashes and inconsistency.
  - Built into the server.
- Semi-synchronous Replication
  - Master commits transaction but waits for at most one slave to acknowledge having received and stored the correspondent event before replying to the client.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Traditional Architectures

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

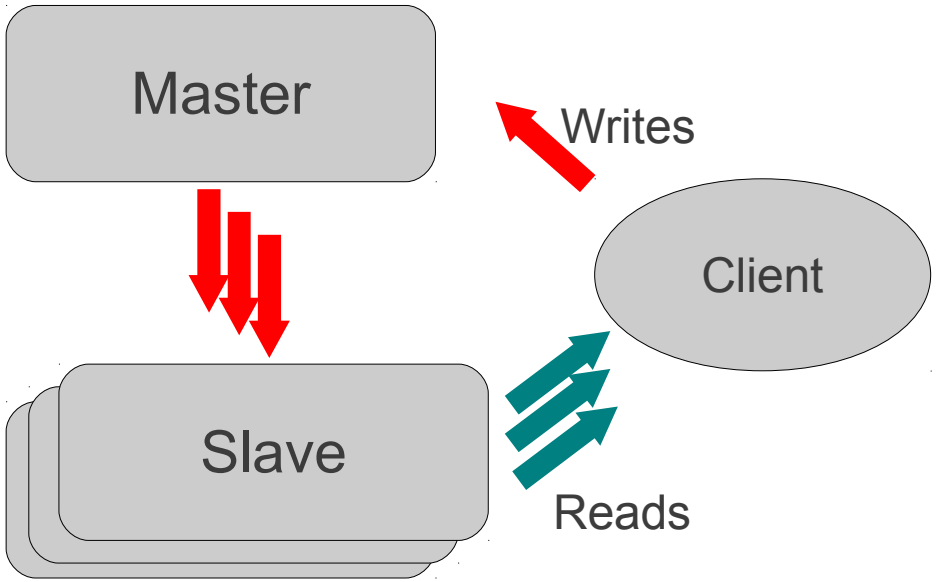
Single Master, Single Slave



- Backups
- Reports
- Disaster Recovery (geographically distant servers)
- Recovering from Human errors (Time-delayed replication)
- Add more slaves seamlessly

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

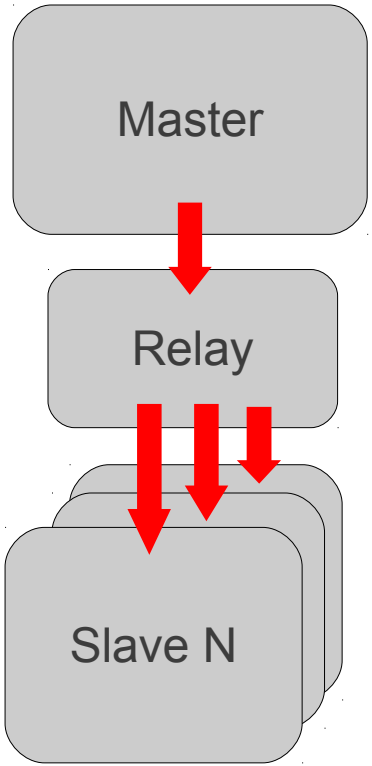
Scaling-out read operations



- Off-loading the master
  - Different type of queries routed to different servers
- Different hardware profiles for master and slaves
  - SSD for enhancing master performance
  - Large RAM / caches to enhance Slaves performance
- Load Balancing
- Query routing policies

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

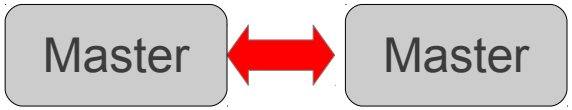
Relay Server

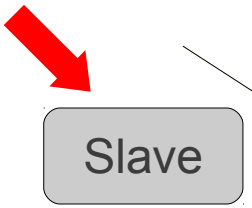


- Relay Slave.
- Blackhole storage engine.
  - No data stored on the relay slave.
- Reduce load at the master.
- Improved master side filtering.
  - Sensitive data can be kept only at one physical server.
- Relay has binary log active.

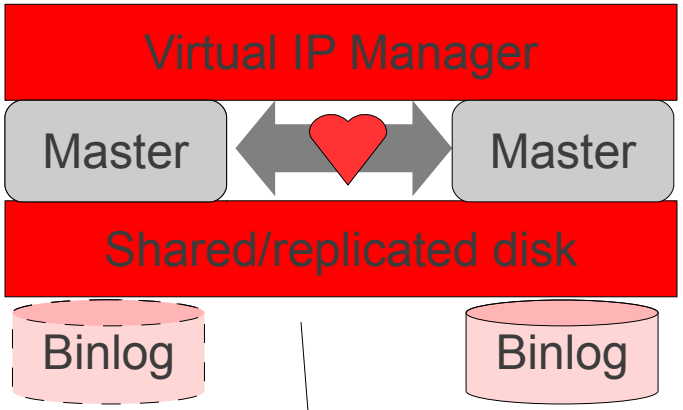
Copyright © 2012, Oracle and/or its affiliates. All rights reserved

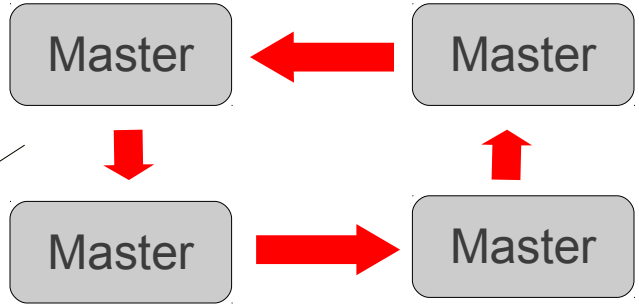
High Availability





Dual Masters





Circular Replication

Active/Passive Shared disk

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

High Availability

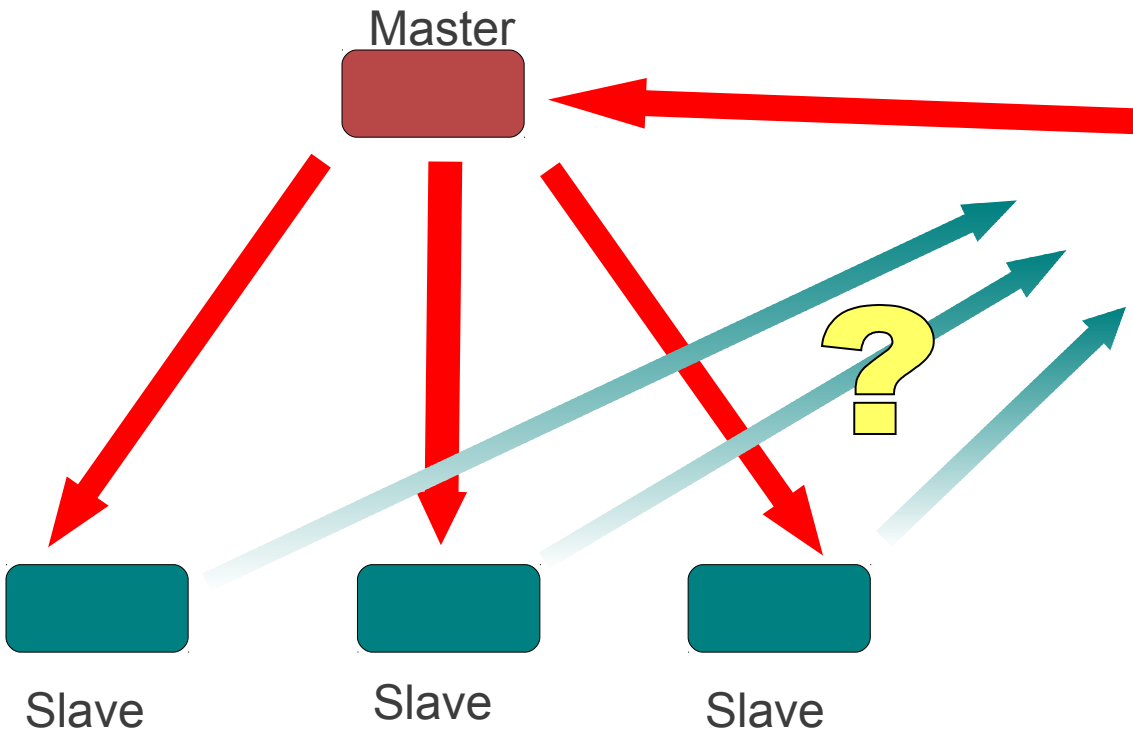
- High Availability: fail-over
  - Servers can crash (hardware, software or even power failure).
  - Services should not.
- Dual masters, circular replication (conflict free partition workload on each).
  - Seamless fail-over of affected partitions
  - Scaling out with slaves.
    - Ready to step up and replace a failed master (on dual masters what out for whether the slave is already ahead of second master – slave promotion instead)
- Active / Passive – binlog is shared by the two master's, on fail-over binlog positions match

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Load Balancing

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Load Balancing



Writes to a master.

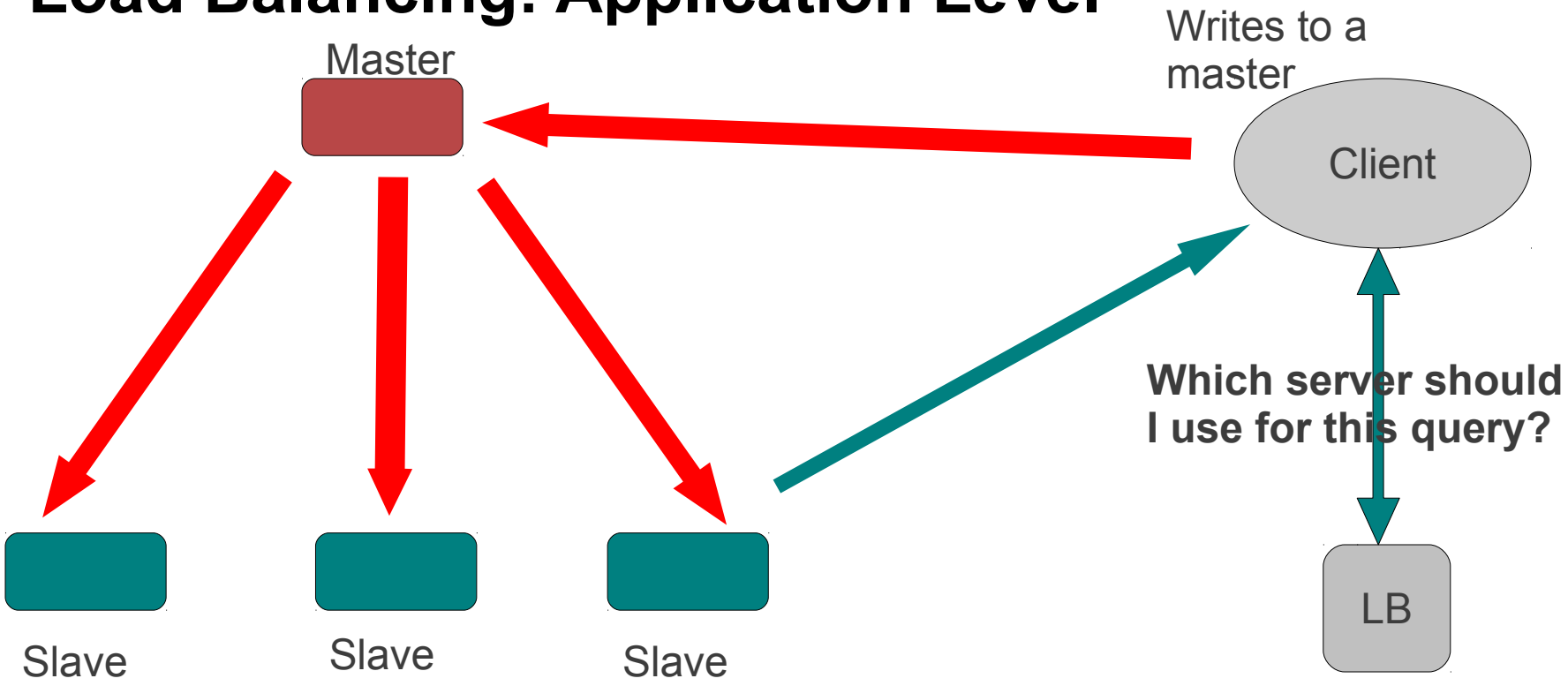
Client

Reads from a pool of slaves.

From which one?

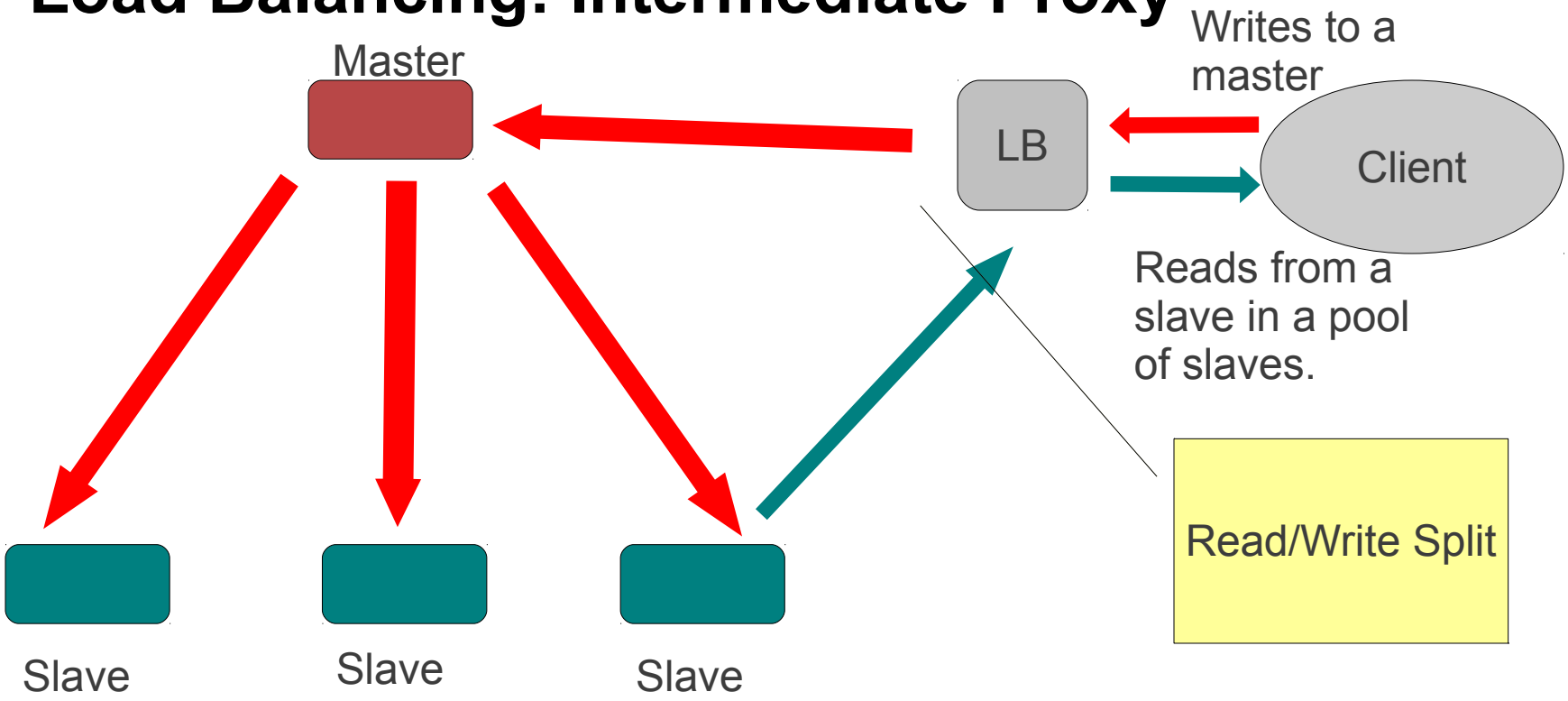
Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Load Balancing: Application Level



Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Load Balancing: Intermediate Proxy



Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Load Balancing

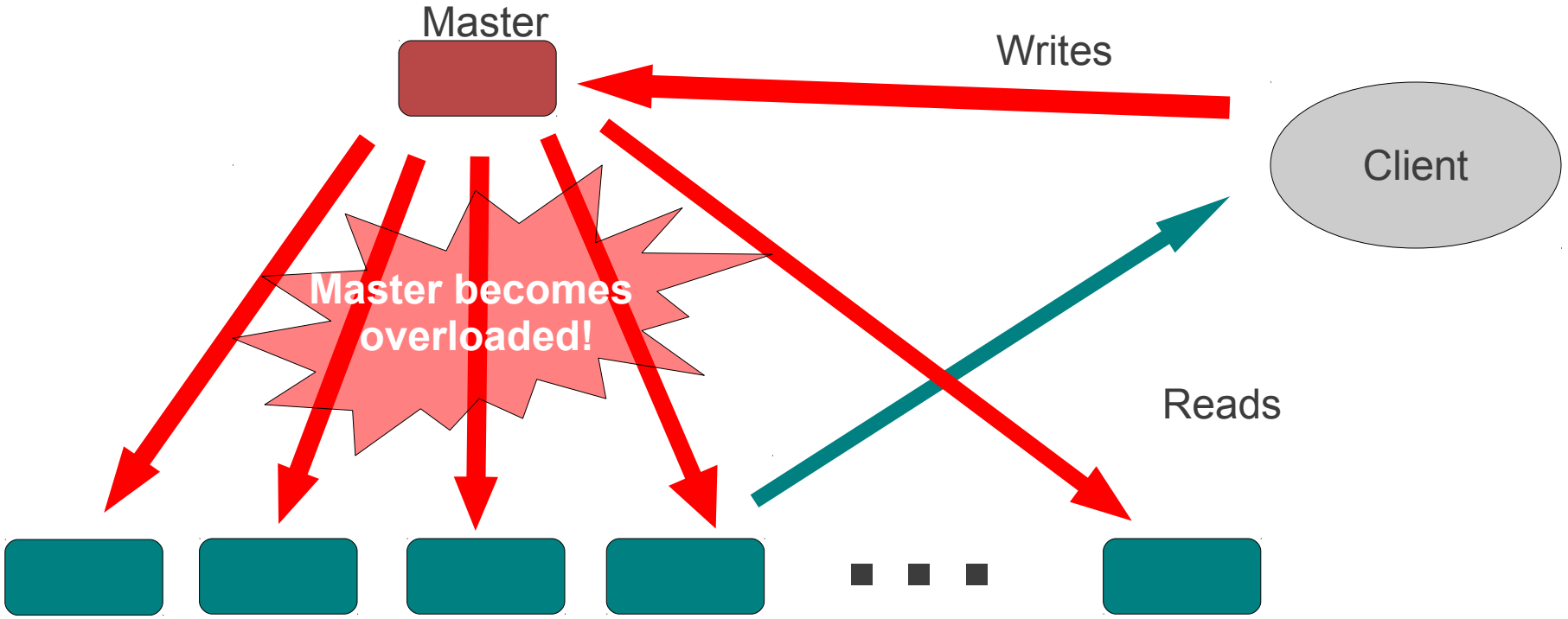
- Requires monitoring tools (status and performance)
  - Servers crash every now and then!
- Requires intelligent query routing (reads vs writes)
  - Which queries go where?
- Session consistency is something to keep in mind!
- Enter MySQL Proxy (alpha quality), PHP mysqlnd, Connector/J …
  - http://dev.mysql.com/downloads/mysql-proxy/
  - http://dev.mysql.com/downloads/connector/php-mysqlnd

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Hierarchical Replication

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

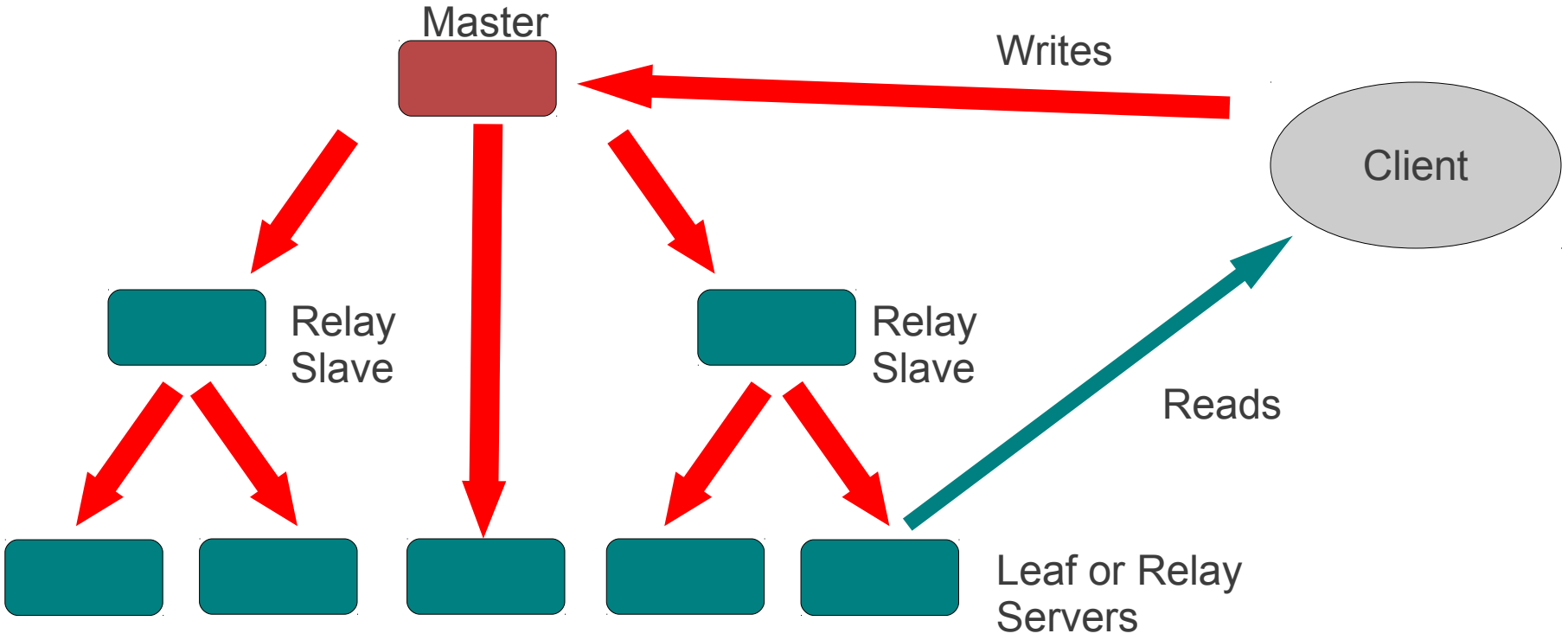
Hierarchical Replication



Hundreds of slaves connected to the master.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Hierarchical Replication



Hundreds of slaves more down the hierarchy ...

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Hierarchical Replication

- Scale-out hundreds of servers
  - Blackhole storage engine on relays.
  - Relay servers have binary logs ON, leaf slaves do not need it.
- Group data domains/partitions
  - Sensitive information routed through parts of the hierarchy.
  - Event filtering.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Hierarchical Replication

- Hierarchy-wide consistency!
  - Asynchronous propagation => writes take time to propagate from master to slaves: reading from a slave may return stale data.
  - Event positions are filename and offset => different on every intermediate slave.
  - Wait for data to propagate.
    - Global transaction identifiers (**provided by the server starting from MySQL 5.6.5 DMR**).
    - Poll each relay server down the chain until data has propagated.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Data Aggregation: Multi-source Replication

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

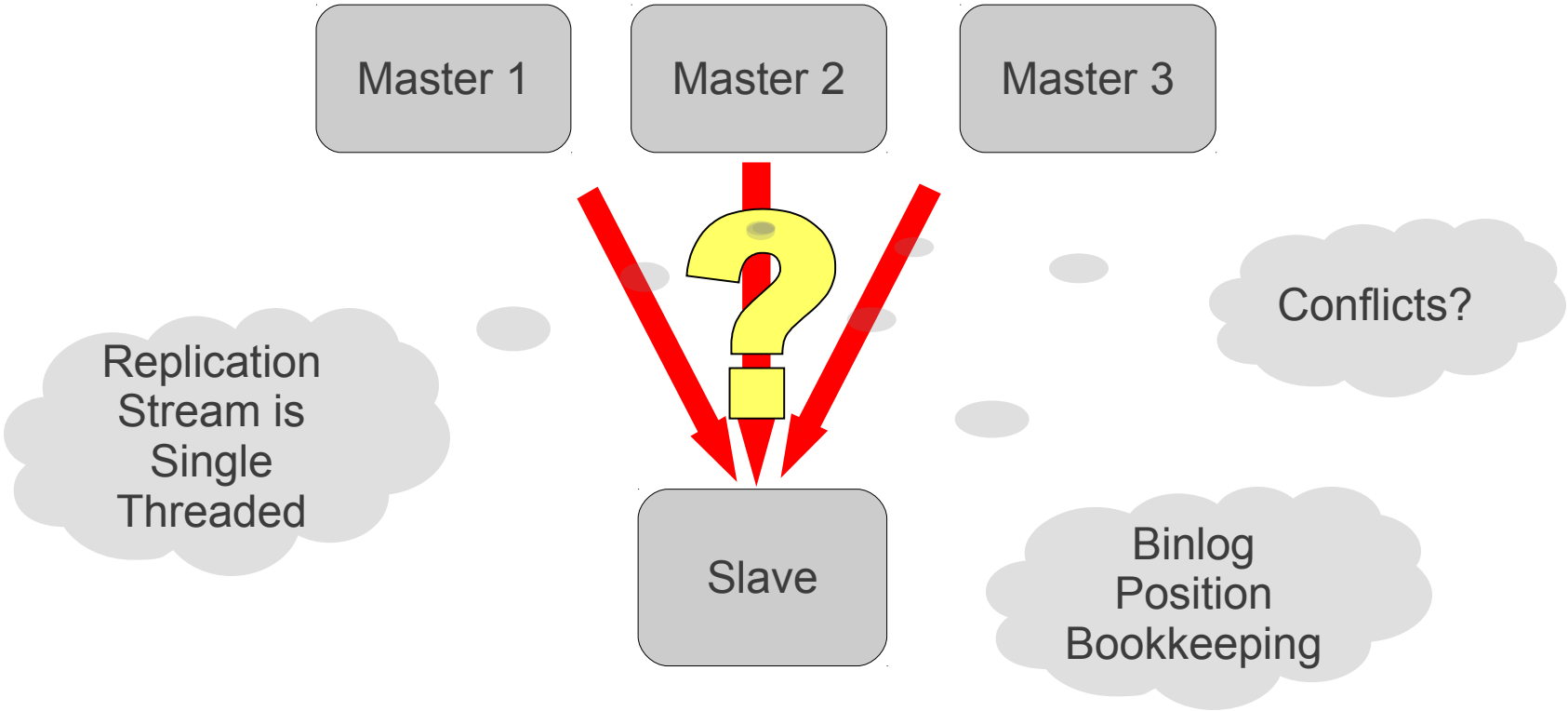
Multi-source Replication

- Why?
  - Aggregate data from different masters, different data-centers, different clusters
- How?
  - Time-share Replication
    - Typically round-robin policy
    - Aggregating data from clusters in different timezones
    - Slave controlled by specific client
    - You can do it all at SQL level
      - http://mysqlmusings.blogspot.pt/2011/04/round-robin-multi-source-in-pure-sql.html

  - Inter-cluster
- Requirements: Conflict free load

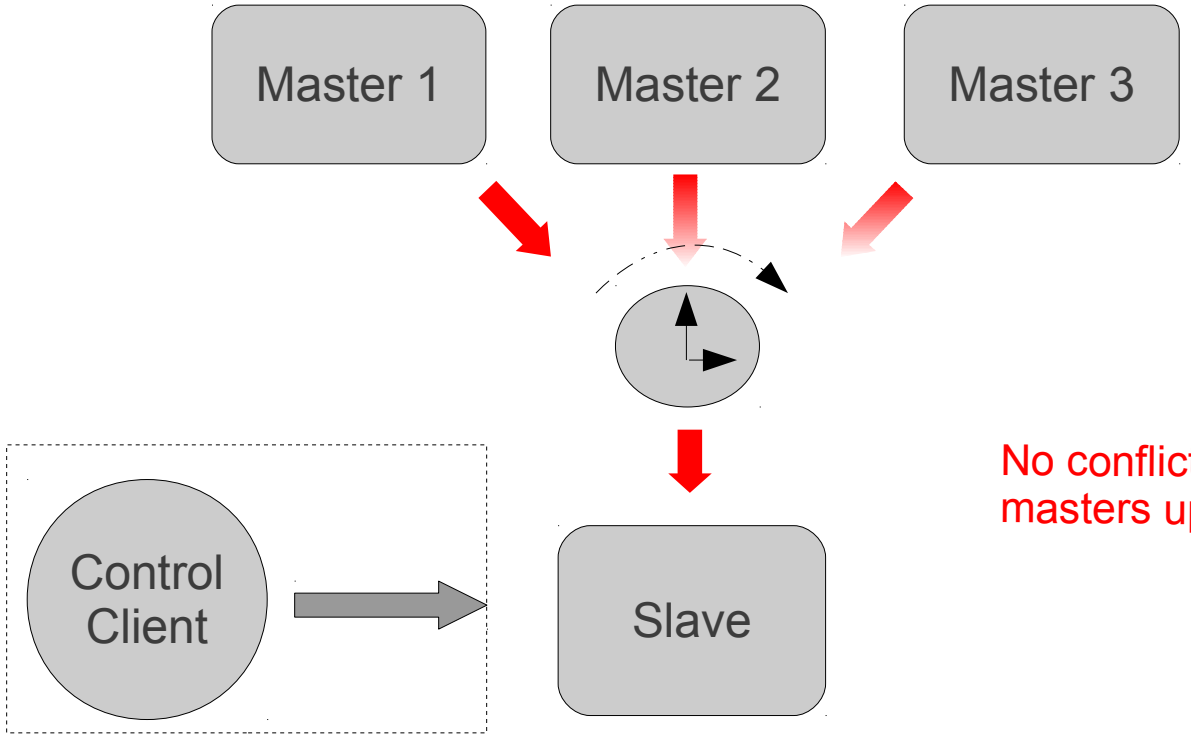
Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Multi-source Replication



Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Multi-source Replication: Time share



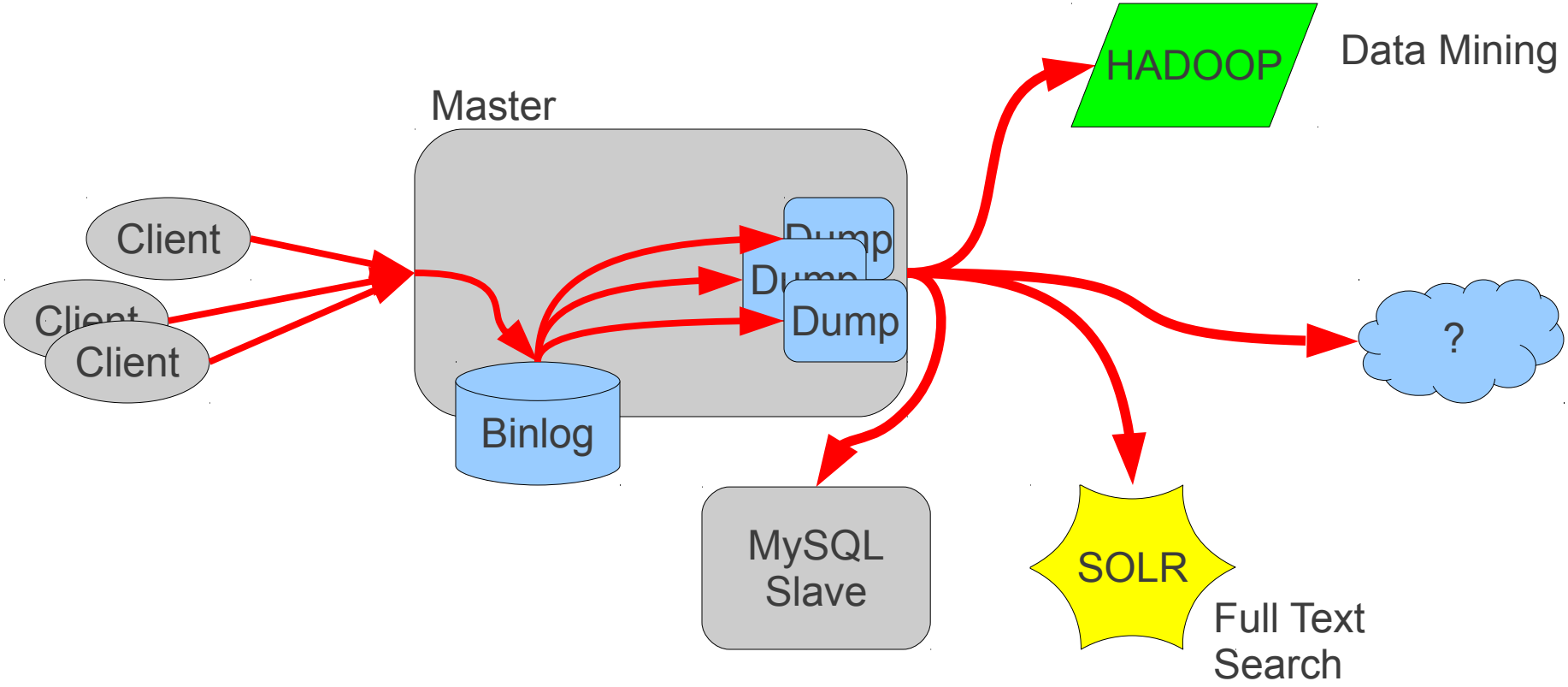
No conflict resolution, so let masters update different data

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Data Integration

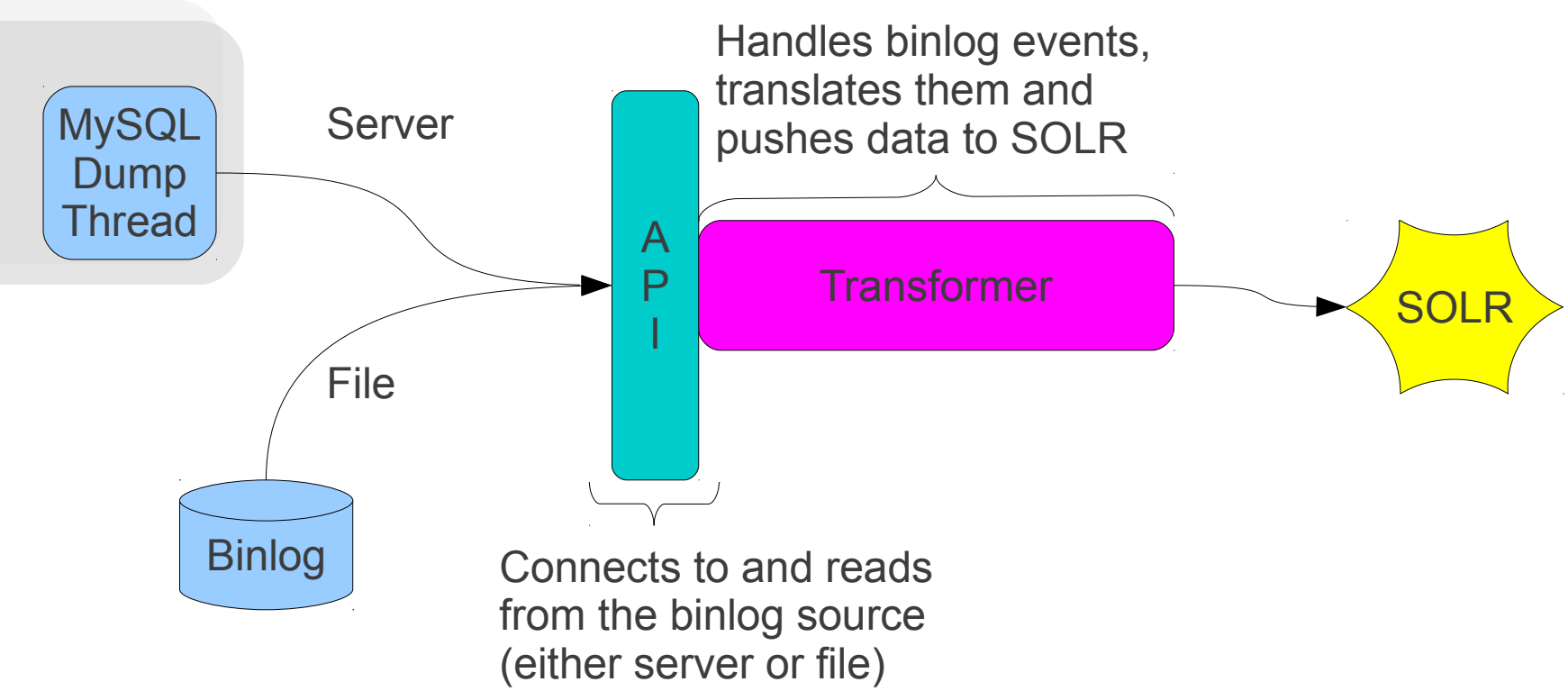
Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Integrate Data Into Other Systems



Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Integrate Data Into Other Systems
via mysql binary log API



Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Integrate Data Into Other Systems

- Library to process replication events
- Application Programming Interface
- Simple, extensible, efficient
  - Just hook a client thread in the wait_for_next_event handler!
- Something you can preview (not GA yet):
  - https://launchpad.net/mysql-replication-listener
  - http://www.oscon.com/oscon2011/public/schedule/detail/18785
  - https://blogs.oracle.com/MySQL/entry/mysql_5_6_replication_new

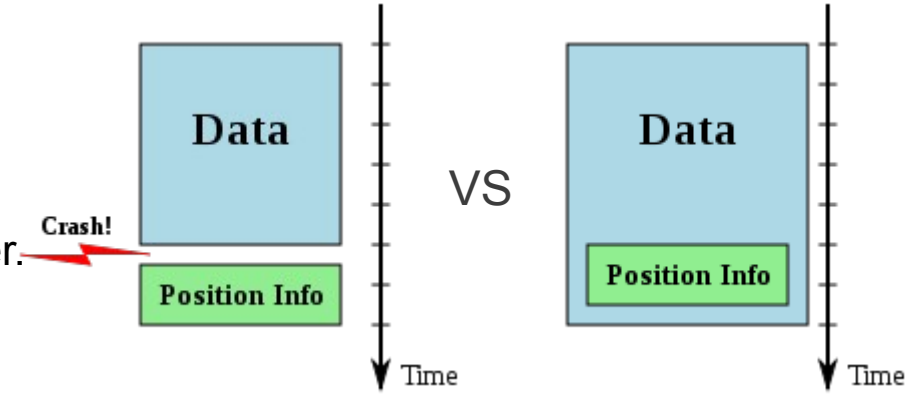
Copyright © 2012, Oracle and/or its affiliates. All rights reserved

MySQL 5.6 and Advanced Replication Architectures Enablers

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Replication Metadata on System Tables

- Crash-safe (replication metadata stored in transactional tables)
  - Robust, highly available setups
- SQL interface to metadata
  - Access data through regular user session.
  - Self-contained: time-shared multi-source easily implemented in SQL and in the server.
  - Make use of special features like MySQL events, stored procedures, triggers, etc...



Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Global Trans Ids: Automatic Failover and more...

- Promote any slave to be the new master.
- Automatically pick replication stream correctly:
  - Slave auto-positioning;
  - Slave automatically skips transactions that already exist in the server.
- Advanced topologies with much more automatic reconfiguration.
  - Session consistency facilitator.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Global Trans Ids: Automatic Failover and more...

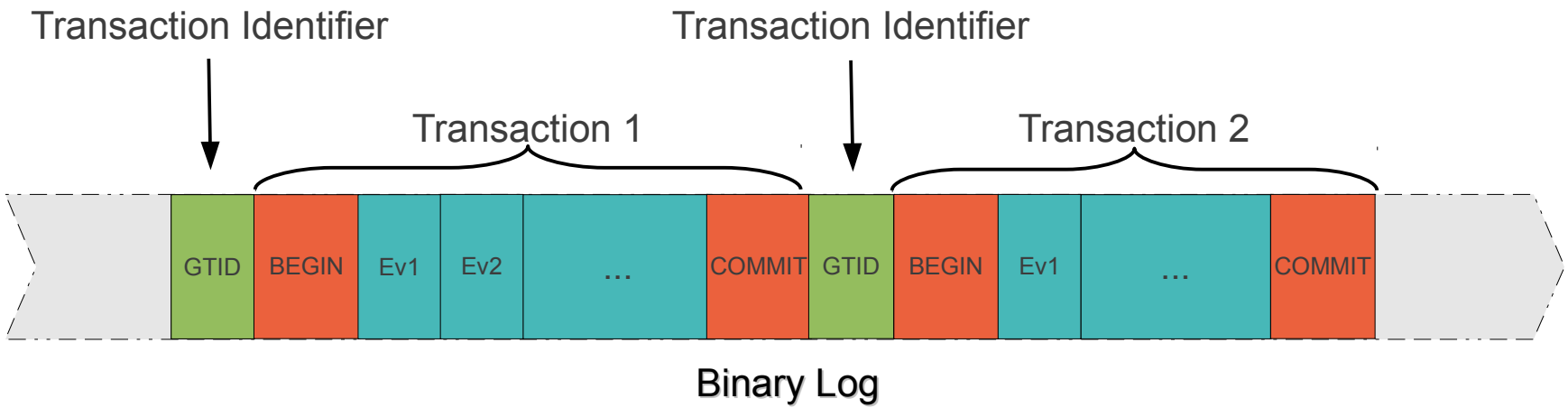
- Logical Identifier: (*Server UUID, Transaction Number);*
  - Human readable/friendly;
  - Uniquely assigned when the transaction executes;
  - Preserved when the transaction is re-applied.
- Server State: set of applied transactions.
  - Enables slave auto-positioning;
  - Failover facilitator;
  - Protects against undesirable transaction re-execution
- Feature designed to work with transactional engines (InnoDB).

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Global Trans Ids: Automatic Failover and more...

- Persistence:
  - It is written to the binary log.
  - Precedes a collection of events that comprise a transaction.

5.6 DMR



Copyright © 2012, Oracle and/or its affiliates. All rights reserved

MySQL Utilities: What is it?

- A collection of Python utilities for managing MySQL databases
- MySQL Workbench Plugin
- Available under the GPLv2 license
- Written in Python
- Python library to grow solutions for common administrative problems
- Download MySQL Workbench from:
  - **http://www.mysql.com/downloads/workbench/**
- You can also download the latest development source code tree for the MySQL Workbench Utilities from:
  - **http://launchpad/net/mysql-utilities**

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

MySQL Utilities: What can I do with it?

- Easily administer MySQL servers from the command line
  - mysqldbcompare – compare databases
  - mysqldbcopy – copy databases between servers
  - **mysqlfailover** – Automatic fail-over
  - **mysqlrpladmin –** General replication administration utility
  - **mysqlrplshow** – show a graph of your topology
  - **mysqlreplicate** – setup replication
  - mysqlrplcheck – check replication configuration
  - ...
- Build your own tools on top of the core of the library, e.g., automate timeshare multi-source replication setup or failing-over to a slave

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

MySQL Proxy

- Simple program in-between client and mysql server
  - Monitor, analyze, transform, load balance
  - Use cases and possibilities are enormous
- Alpha Release Available at:
  - **http://dev.mysql.com/downloads/mysql-proxy/**
- Documentation available at:
  - **http://dev.mysql.com/doc/refman/5.5/en/mysql-proxy.html**

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

The Binary Log API

- A C++ library used for connecting to a MySQL server and process the replication stream as a slave.
- Enables complex setups and data streaming to heterogeneous slaves.
- Not GA.
- Source code is publicly available at (licensed under GNU GPL v2):
  - **https://launchpad.net/mysql-replication-listener**

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Summary

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Summary

- MySQL replication concepts and components: replication is simple
- Popular MySQL replication use case scenarios: replication is flexible and versatile
- Scenarios that take MySQL replication to a whole new more advanced level:
  - Load Balancing
  - Hierarchical replication
  - Data replication and aggregation
  - Data integration
- MySQL features that are enablers for more advanced replication scenarios
  - Including some of the shiny new features included in the latest MySQL 5.6 DMRs.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Q&A

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved

Hardware and Software

ORACLE

Engineered to Work Together

Copyright © 2012, Oracle and/or its affiliates. All rights reserved


ORACLE®