facebook

MySQL at Facebook

Yoshinori Matsunobu
Production Engineer, Facebook
Oct 2, 2017 – Oracle Open World

MySQL teams at Facebook

- Production Engineering (SRE/PE)
  - MySQL Production Engineering
  - Data Performance
  - Based in Menlo Park and London

- Software Engineering (SWE)
  - MySQL Engineering
    - MyRocks
    - Others (Replication, Client, InnoDB, etc)
  - Based in Menlo Park and Seattle

User Database (UDB) at Facebook

- Storing Social Graph
- Massively Sharded
- Low latency
- Automated Operations
- Pure Flash Storage (Constrained by space, not by CPU/IOPS)

MySQL at Facebook

- We upgraded from 5.1 to 5.6 in 2013-2014
- Currently based on MySQL 5.6
- Created many features on top of 5.6, to support our services
- We started working with Oracle to take our patches into official MySQL
- Started evaluating MySQL 8.0

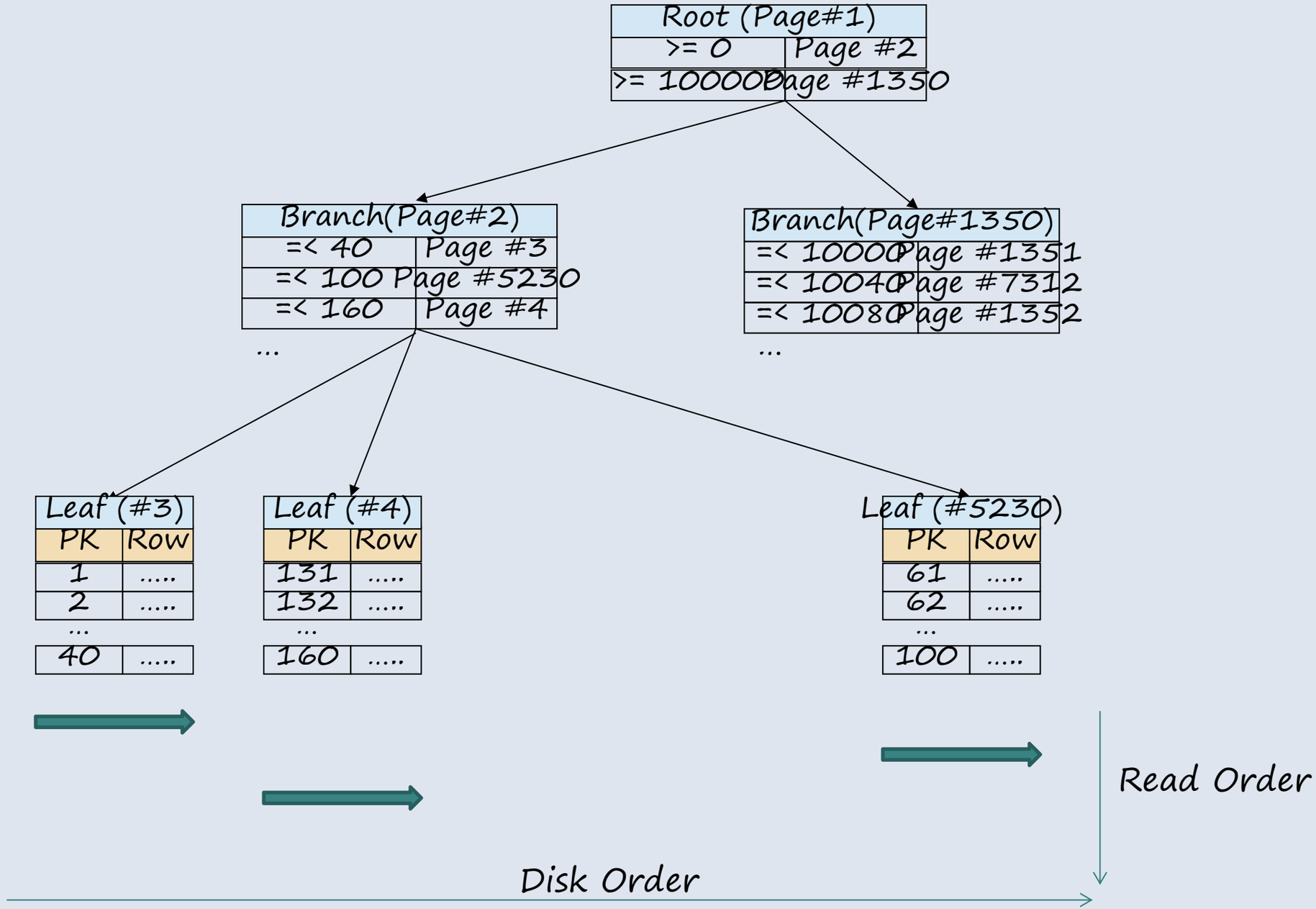
InnoDB Online Defragmentation

- Added a new command
  - ALTER TABLE table_name DEFRAGMENT INDEX index_name
- Merging nearby fragmented pages and freeing up empty pages
- Persisted defragmentation statistics in mysql.innodb_index_stats
  - Saved up to 20% used space
  - Writing more data to storage was negative impact

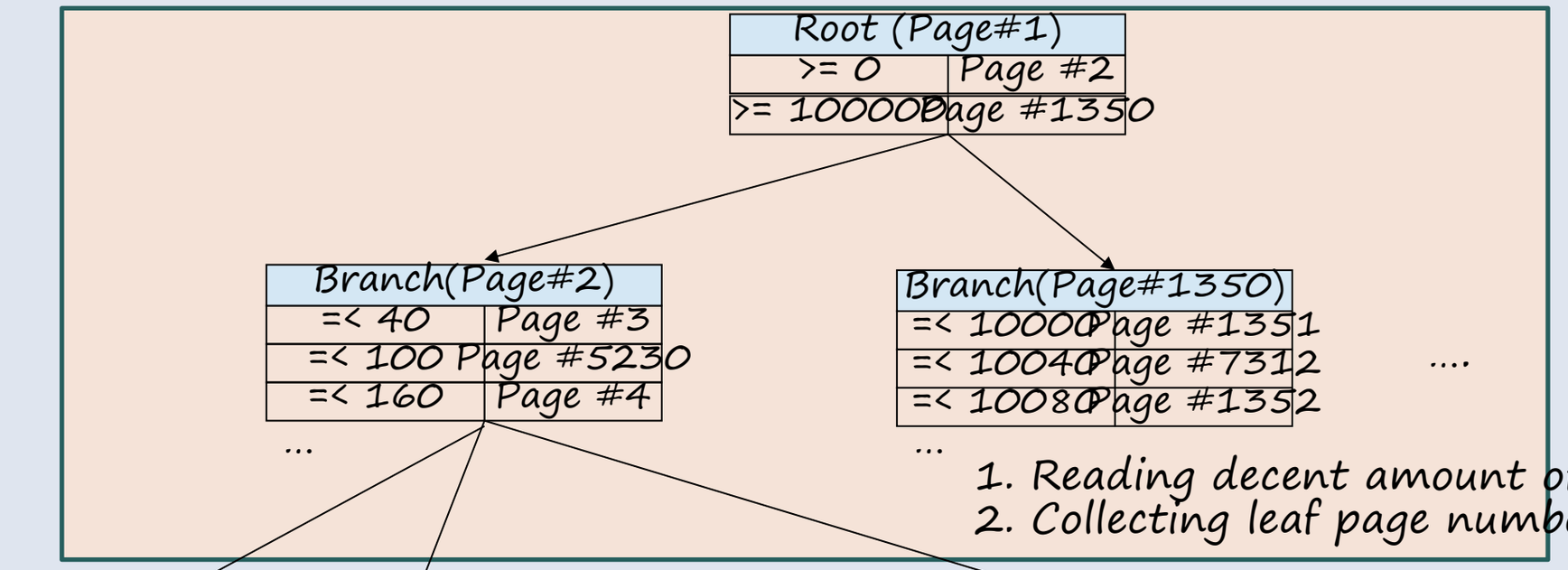
Reducing writes to InnoDB Doublewrite

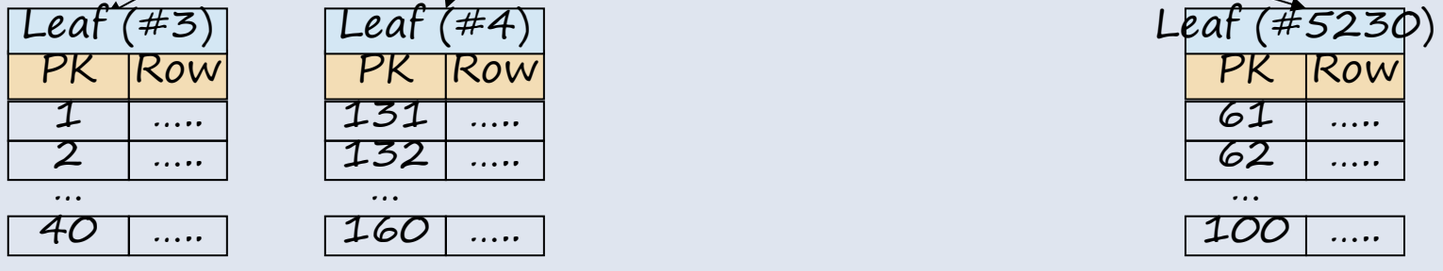
- Double write buffer is helpful to avoid torn-page problem
- But write volume becomes 2x or even more
  - Writing the same number of pages as data files
  - Writing 16KB per page regardless of compression
- Added "innodb_doublewrite=2" for "detecting" torn pages
  - Writing only page ids to doublewrite buffer (4B instead of 16KB)
  - On crash recovery, checking target pages are corrupted or not
  - In many cases, pages are not corrupted
  - If pages are corrupted, reimaging from other slaves

Improving full table scan in InnoDB



InnoDB Logical Read-Ahead





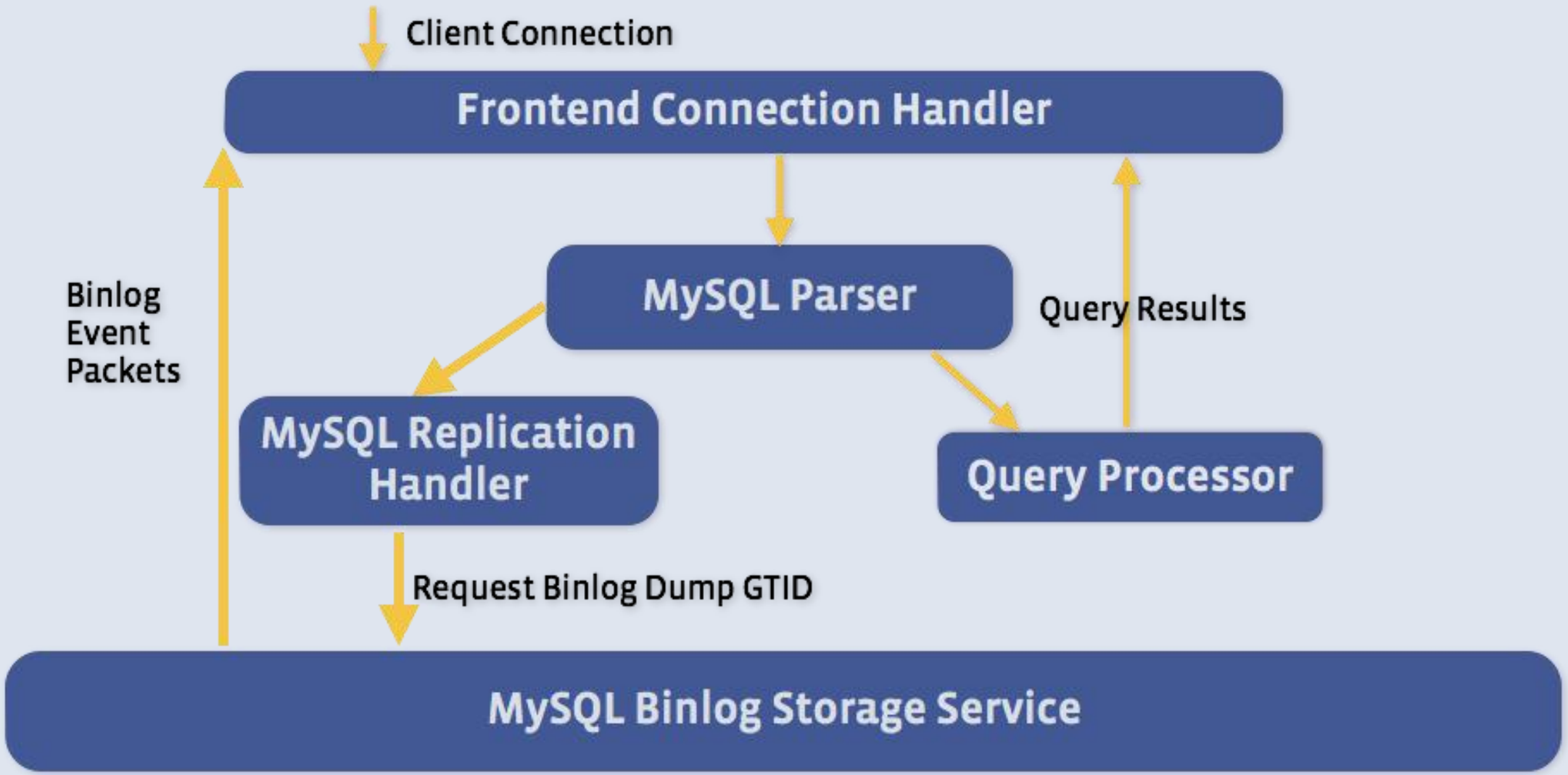
3. Reading from disk by leaf page # order (loaded into InnoDB buf pool)
4. Reading rows by primary key order (no random disk read penalty since pages are in bu

Got 10x faster full table scan throughput on typical fragmented tables

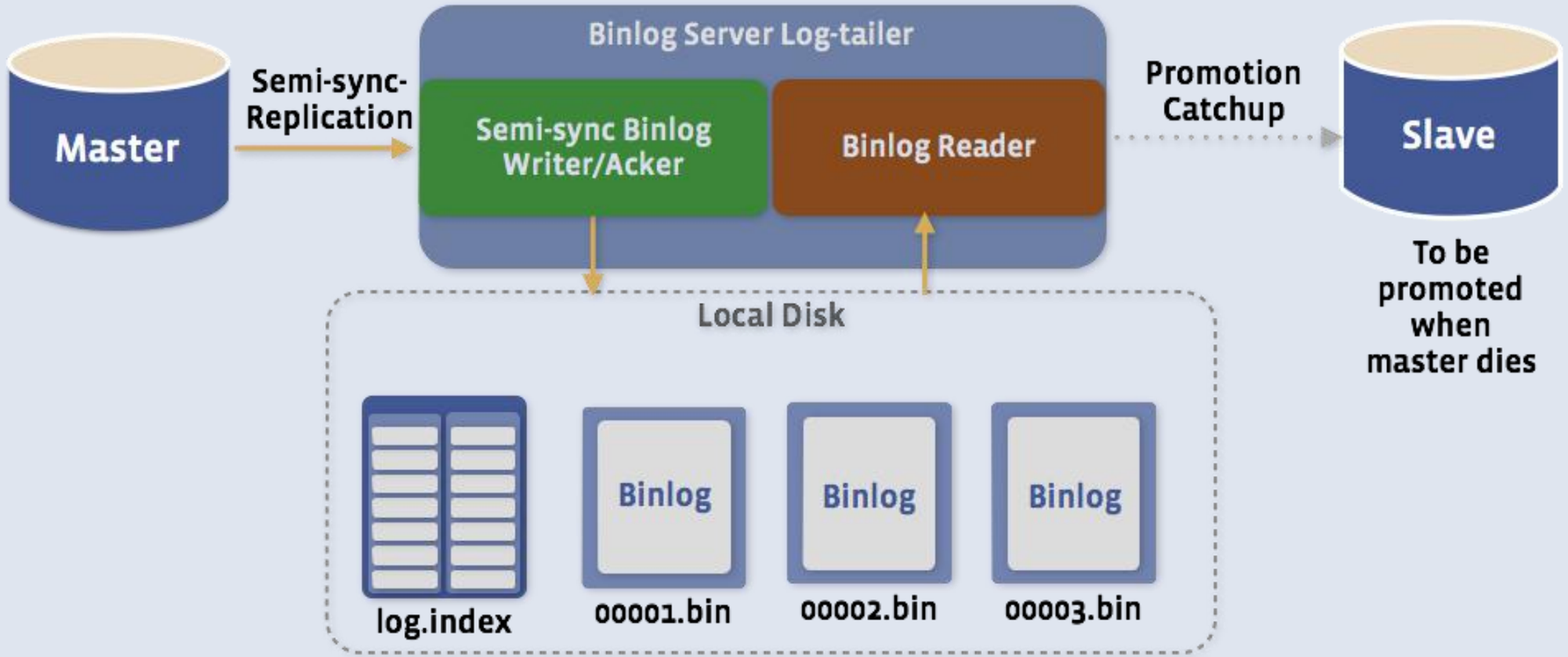
Other important Facebook extensions

- Asynchronous MySQL client
- Query throttling based on number of active threads
- Super read-only
- Per user default session variables
- Crash safe slave/master when using GTID and Multi-Threaded Slave with less durability
- mysqlbinlog speaking Semisync protocol
- START TRANSACTION WITH CONSISTENT $ENGINE SNAPSHOT, returning consistent binlog state

Binlog Server

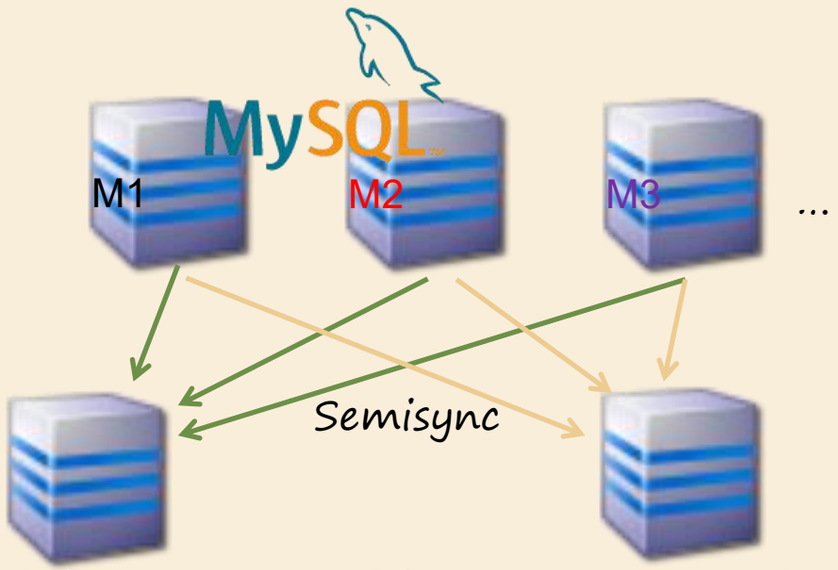


Binlog Server with Loss-Less Semisync



Single copy per region, with Binlog Server

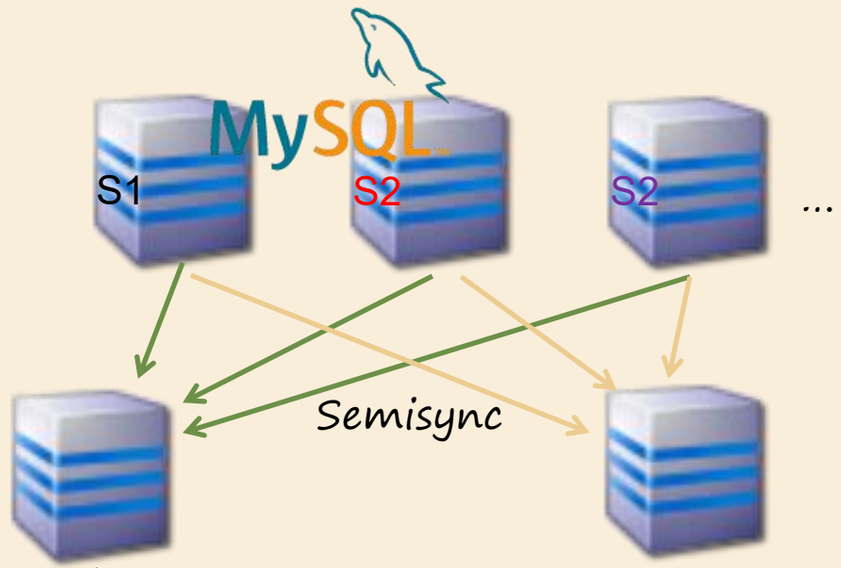
DC1 (primary master region)



Binlog Servers with Loss-Less Semisync

Async Repl

DC2 (secondary master region)



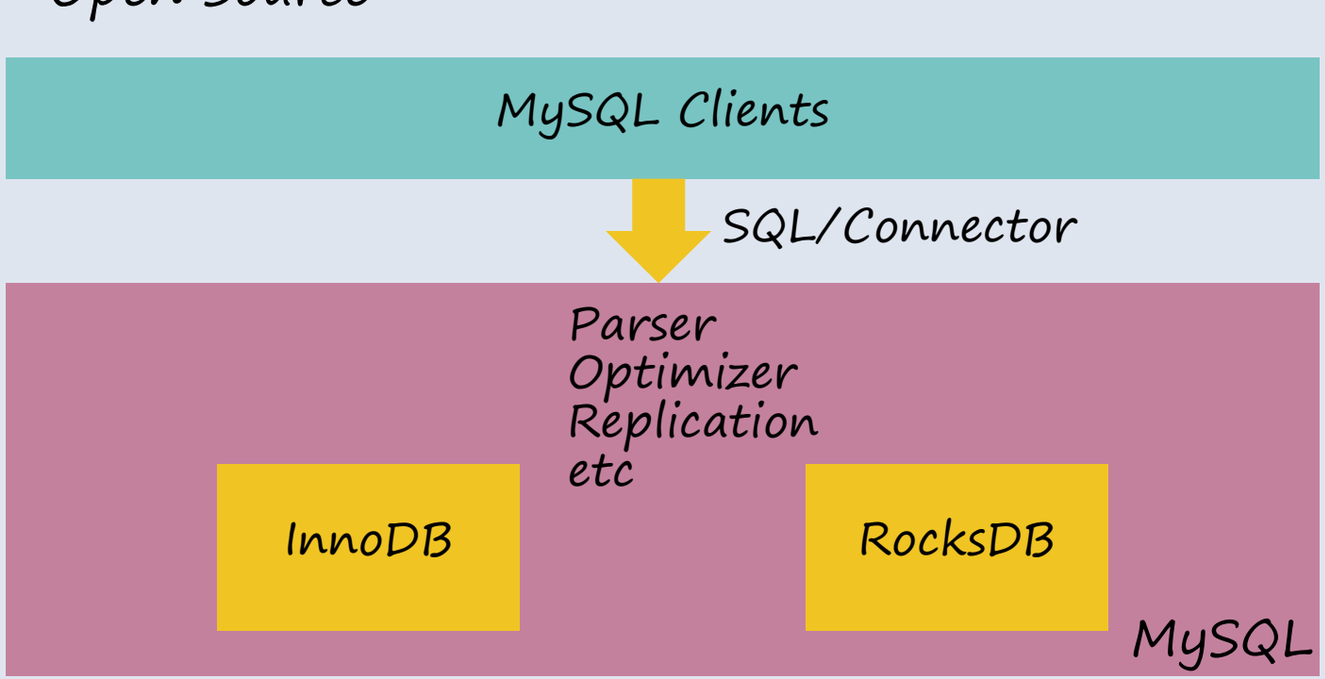
Binlog Servers with Loss-Less Semisync

- We created a "Binlog Server" that speaks semisync replication protocol and stores binary logs locally
- No dedicated local semisync slave is needed (Single Copy per Region)

DC3

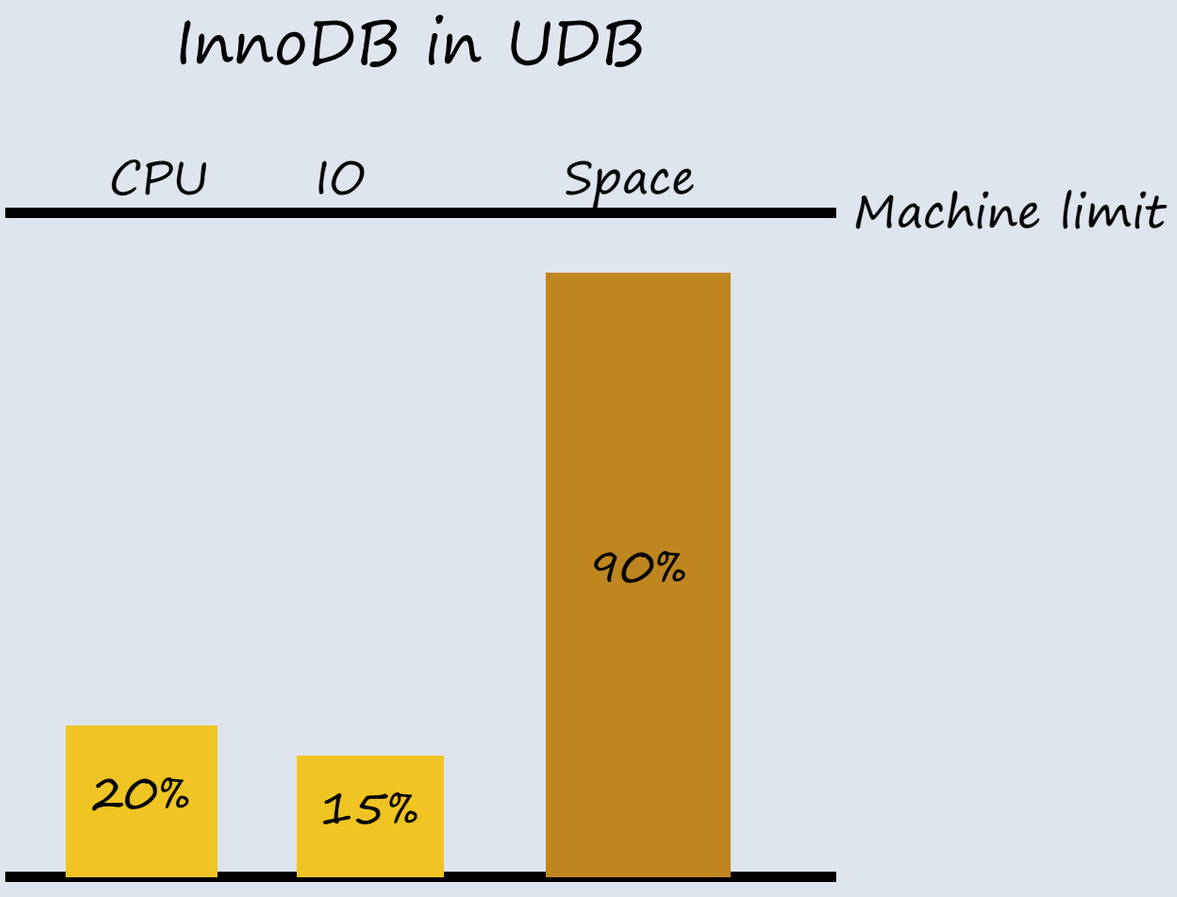
What is MyRocks

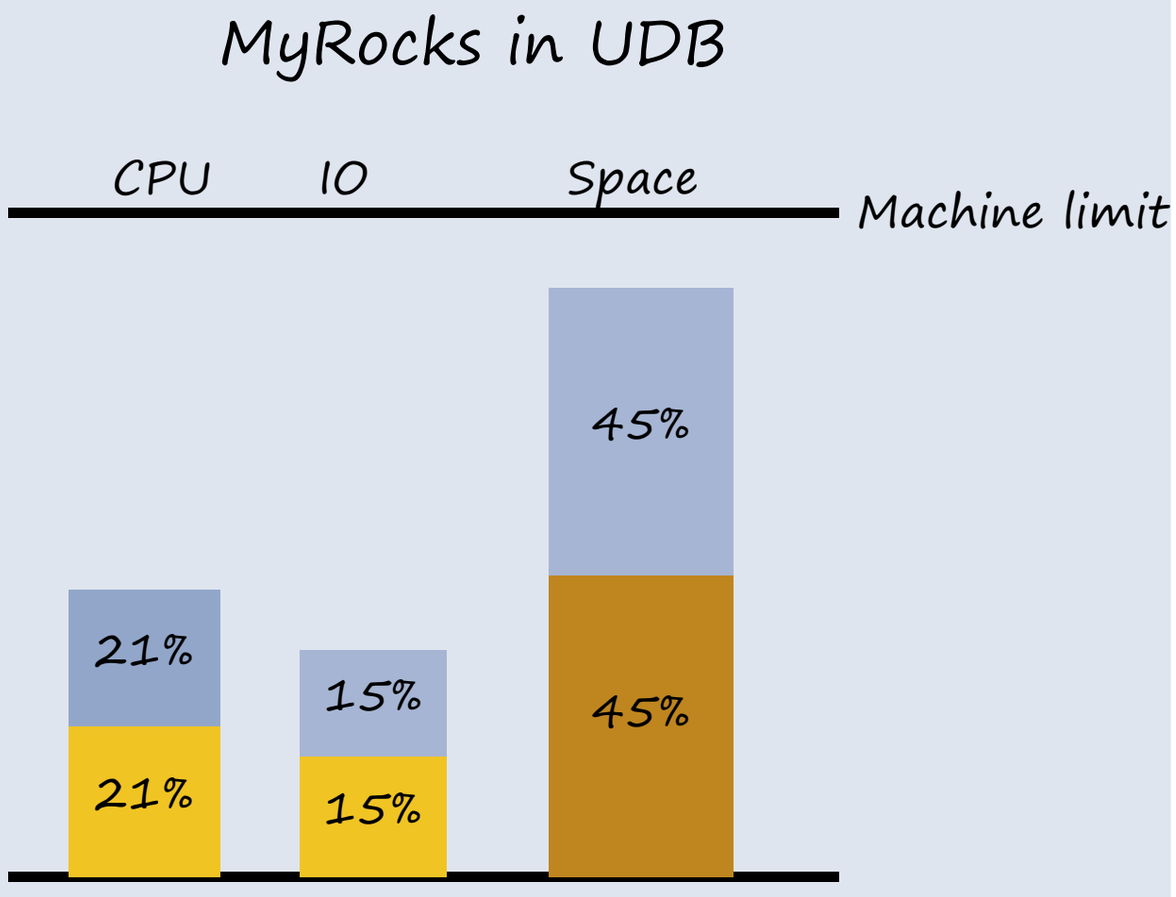
- MySQL on top of RocksDB (RocksDB storage engine)
- Taking both LSM advantages and MySQL features
  - LSM advantage: Smaller space and lower write amplification
  - MySQL features: SQL, Replication, Connectors and many
- Open Source



http://myrocks.io/

MyRocks Initial Goal at Facebook





Migration in Production

- Continuous data consistency check between InnoDB and MyRocks
- Shadow traffics tests
- Deployed on slaves
- Deployed on masters

Towards MySQL 8.0

- We started evaluating MySQL 8.0
- We started working with Oracle to take our patches in official MySQL
- Planning to use Performance Schema and to drop Table Statistics patches
- Transactional DDL and better optimizer statistics are great for us

Future Plans

- Helping Oracle to take our patches
- MySQL 8.0 Deployment
- Supporting multiple storage engines
- Supporting "Bigger Small Data"

facebook