MyRocks deployment at Facebook and Roadmaps

Yoshinori Matsunobu
Production Engineer / MySQL Tech Lead, Facebook
Feb/2018, #FOSDEM #mysqldevroom
Agenda

- MySQL at Facebook
- MyRocks overview
- Production Deployment
- Future Plans
MySQL “User Database (UDB)” at Facebook

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

- MySQL on top of RocksDB (RocksDB storage engine)
- Open Source, distributed from MariaDB and Percona as well

MySQL Clients

- SQL/Connector
- Parser
- Optimizer
- Replication etc

InnoDB

- RocksDB

MySQL

http://myrocks.io/
MyRocks Initial Goal at Facebook

InnoDB in main database

- CPU: 20%
- IO: 15%
- Space: 90%
- Machine limit: 20%

MyRocks in main database

- CPU: 21%
- IO: 15%
- Space: 45%
- Machine limit: 15%
MyRocks features

- Clustered Index (same as InnoDB)
- Bloom Filter and Column Family
- Transactions, including consistency between binlog and RocksDB
- Faster data loading, deletes and replication
- Dynamic Options
- TTL
- Online logical and binary backup
MyRocks vs InnoDB

**MyRocks pros**
- Much smaller space (half compared to compressed InnoDB)
  - Gives better cache hit rate
- Writes are faster = Faster Replication
- Much smaller bytes written (can use more affordable flash storage)

**MyRocks cons (improvements in progress)**
- Lack of several features
  - No SBR, Gap Lock, Foreign Key, Fulltext Index, Spatial Index support. Need to use case sensitive collation for perf
- Reads are slower, especially if your data fits in memory
- More dependent on filesystem and OS. Lack of solid direct i/o. Must use newer 4.6 kernel
- There are too many tuning options beyond buffer pool, such as bloom filter, compactions etc
Creating first MyRocks instance without downtime

- Picking one of the InnoDB slave instances, then starting logical dump and restore

- Stopping one slave does not affect services
Faster Data Loading

Normal Write Path in MyRocks/RocksDB

- Write Requests
  - WAL
  - MemTable
  - Level 0 SST
  - Level 1 SST
  - Level max SST
  - Flush
  - Compaction

Faster Write Path

- Write Requests
  - Level max SST

“SET SESSION rocksdb_bulk_load=1;”
Original data must be sorted by primary key
Creating second MyRocks instance without downtime

- Master (InnoDB)
- Slave1 (InnoDB)
- Slave2 (InnoDB)
- Slave3 (MyRocks)
- Slave4 (MyRocks)

*myrocks_hotbackup (Online binary backup)*
Promoting MyRocks as a master
Promoting MyRocks as a master
Our current production status

We **COMPLETED** InnoDB to MyRocks migration in UDB

We saved **50% space** in UDB compared to compressed InnoDB

We started working on migrating other large database tiers
Development Roadmaps

- Helping MariaDB and Percona Server to release with stable MyRocks

- Matching read performance vs InnoDB
  - https://smalldatum.blogspot.com

- Supporting Mixed Engines

- Better Replication

- Supporting Bigger Instance Size
Mixed Engines

- Currently our production use case is either “MyRocks only” or “InnoDB only” instance

- There are several internal/external use cases that want to use InnoDB and MyRocks within the same instance, though single transaction does not overlap engines

- Online logical/binary Backup support and benchmarks are concerns

- Current plan is extending xtrabackup to integrate myrocks_hotbackup

- Considering to backporting gtid_pos_auto_engines from MariaDB
Better Replication

- Removing engine log
  - Both internal and external benchmarks show that qps improves significantly with binlog disabled
  - Real Problem would be two logs - binlog and engine log, which requires 2pc and ordered commits
  - One Log - use one log as the source of truth for commits -- either binlog, binlog-like service or RocksDB WAL
  - We heavily rely on binlogs (for semisync, binlog consumers), TBD is how much perf we gain by stopping writing to WAL

- Parallel replication apply
- Batching
- Skipping using transactions on slaves
Supporting Bigger Instance Size

- Problem Statement: Shared Nothing database is not general purpose database
  - MySQL Cluster, Spider, Vitess
  - Good if you have specific purposes. Might have issues if people lack of expertise about atomic transactions, joins and secondary keys

- Suggestion: Now we have 256GB+ RAM and 10TB+ Flash on commodity servers. Why not run one big instance and put everything there?

- Bigger instances may help general purpose small-mid applications
  - They don’t have to worry about sharding. Atomic trans, joins and secondary keys just work
  - e.g. Amazon Aurora (supporting up to 60TB instance)
Future Plans to support Bigger Instance

- Parallel transactional mysqldump
- Parallel Query
  - e.g. how to make mysqldump finish within 24 hours from 20TB table?
- Parallel binary copy
  - e.g. how quickly can we create a 60TB replica instance in a remote region?
- Parallel DDL, Parallel Loading
- Resumable DDL
  - e.g. if the DDL is expected to take 10 days, what will happen if mysqld restarts after 8 days?
Future Plans to support Bigger Instance (2)

- Better join algorithm
- Much faster replication
- Can handle 10x connection requests and queries
- Good resource control
- H/W perspective: Shared Storage and Elastic Computing Units
- Can scale read replicas from the same shared storage
Summary

- We finished deploying MyRocks in our production user database (UDB)
- You can start deploying slaves, with consistency check
- We have added many status counters for instance monitoring
- More interesting features will come this year