MySQL: Scaling & High Availability

Production experience for the last decade(s)

Peter Zaitsev, CEO, Percona
February 3rd, 2018

FOSDEM
Let's go to the start of my MySQL story

Going back almost 2 decades
1999
My First MySQL Experience
MySQL in 1999

MySQL 3.23 Alpha

MyISAM storage engine just introduced

Michael “Monty” Widenius personally reply to most MySQL issues
MySQL Challenges in 1999

MyISAM Table Locks

Frequent Crashes (using Alpha software)

2GB File Size Limit on Linux

MyISAM Table Checks and Repairs take very long time
MySQL “Tricks” in 1999

- Sharding for Scalability
- Single Node Sharding to avoid Table Locks
- Building Lots of Summary Tables
MySQL in 2000

MySQL goes Open Source, GPL License
MySQL in 2001

MySQL 3.23 is “Stable”

MySQL 3.23-Max with InnoDB (and BDB) Storage Engines

Initial release of MySQL Replication

MySQL is sued by Progress/Nusphere, Countersues
MySQL Challenges in 2001

Stabilizing InnoDB

Making initial release of MySQL Replication to work
2002

I Joined MySQL AB

Did a little bit of Development

Moved to do Support and Consulting

Manager “High Performance Group”
MySQL Challenges in 2002

Scaling MySQL for Web 2.0

Query Optimization

MySQL Sharding is becoming common

bugs.mysql.com
MySQL in 2003

- MySQL 4.0 is released
- Improved Replication
- Query Cache
LiveJournal “Reference Architecture”
MySQL Tricks in 2003

Memcached

Massive Replication
MySQL in 2004

MySQL 4.1 is available

First “Checkbox” Release

Subqueries and Prepared Statements

MySQL Cluster (NDB) is first available
8:13 pm **MySQL: What do do without hash join?**

I've recently seen number of cases where MySQL nested loops join execution method does not work well at all. What if you do join of large tables, having millions of row each, ie "customers" and "orders"?

In OLTP applications you normally do not bother as you have just work query one or few rows in the main table, while in decision support systems and data mining you might need to fetch million of rows from each table.

Nested Loops are just not going to work in such case. Imagine you're going over 10.000.000 "customers" and joining them to "orders", having about 10 orders per customer in average:

```
SELECT CUSTOMER.NAME, ORDERS.PRICE FROM ORDERS,CUSTOMER WHERE CUSTOMER.COUNTRY="US" and ORDER.CUSTOMER_ID=CUSTOMER.ID;
```

Most other database engines will select HASH JOIN in such case and resolve the query via two table scan and some temporary "hashing" area, while MySQL does not have such choice yet.
MySQL in 2005

MySQL 5.0

Second “Checkbox” Release

Stored Procedures, Views, Triggers

Oracle Acquires Innobase (creator of Innodb)

First Puppet Release
MySQL in 2006

- Scrambling with Innobase Acquisition fallout
- Buys “Netfrastracture”
- Jim Starkey (Firebird founder) Joins MySQL
- “Falcon” Storage Engine
- Hadoop is First Available
2006 – Started MySQL Performance Blog

June 9, 2006

Why MySQL could be slow with large tables?

If you’ve been reading enough database related forums, mailing lists or blogs you probably heard complains about MySQL being unable to handle more than 1,000,000 (or select any other number) rows by some of the users. On other hand it is well known with customers like Google, Yahoo, LiveJournal, TechnoCarat. MySQL has installations with many billions of rows and delivers great performance. What could be the reason?

The reason is normally table design and understanding inner works of MySQL. If you design your data wisely considering what MySQL can do and what it can’t you will get great performance if not, you might become upset and become one of those bloggers. Note - any database management system is different in some respect and what works well for Oracle, MS SQL, PostgreSQL may not work well for MySQL and other way around. Even storage engines have very important differences which can affect performance dramatically.

The three main issues you should be concerned if you’re dealing with very large data sets are Buffers, Indexes and Joins.

[read more...]
Started Percona with Vadim Tkachenko

- Performance Consultants
- Helping Companies to Scale MySQL
MySQL in 2008

MySQL 5.1 is released
Partitioning and Row Based Replication
Sun Microsystems Acquires MySQL AB
Amazon Web Services EC2 Available as GA
Percona in 2008

• Helping Customers requires writing code
• Percona XtraDB (fork of Innodb)
• 2nd edition of High Performance MySQL Book
MySQL in 2009

- Oracle Acquires Sun and so MySQL
- "Oracle will kill MySQL" rumors
- MariaDB started by Michael "Monty" Widenius as MySQL Alternative
- Amazon RDS MySQL is available
- Initial version of MongoDB released
MySQL in 2010

MySQL 5.5

Scalability, Performance Schema

OpenStack Initial Release
Percona in 2010

Percona Server 5.1

Percona Xtrabackup
MySQL Challenges 2010

- Scaling MySQL with CPU Cores
- MySQL Deployment Automation
- Getting MySQL Ready for Cloud
- Automated Replication Failover
2012

- Percona XtraDB Cluster 5.5 is Available
- New Generation, Cloud Friendly High Availability Technology for MySQL
MySQL in 2013

- MySQL 5.6 is available
- Better Scalability, Performance Schema, GTID, Optimizer
- SSDs are mainstream
- Initial release of Docker
Amazon RDS Aurora is available
MySQL in 2015

MySQL 5.7 is available

Scalability, JSON, Document Store, Parallel, Multi Source Replication

ProxySQL is Available

Percona Acquires Tokutek (TokuDB)
MySQL Challenges 2015

- NoSQL Solutions, no need for Manual Sharding
- Schema Changes are Painful
- Many Developers do not understand or like SQL
- MySQL Can’t use multiple CPU cores for a query
- MySQL single thread performance tend to get worse for single queries
Better Performance at Scale

Sysbench: OLTP_RO Point-Selects

2.1x Faster than MySQL 5.7
2.8x Faster than MySQL 5.6
Sorry state of MySQL single thread performance

http://bit.ly/2oMvu2a
MySQL Ecosystem in 2016

- Initial Release of Percona Monitoring and Management
- ClickHouse Open Sourced by Yandex
MySQL in 2017

- MySQL Group Replication
- MySQL InnoDB Cluster
MySQL 2018 (to date)

- MySQL 8.0.4 Release Candidate Available
- MyRocks available with Percona Server 5.7
MySQL in 2018

Is not the “only” Open Source Database Any More

Still #1 Open Source Relational Database

Used in Combination with Other Technologies in Data Architectures

Commonly Deployed in the Cloud

All major Cloud Providers have MySQL Compatible DBaaS
Modern MySQL Scalability
Single MySQL Instance Can Do

- Hundreds of Thousands of Queries/Sec
- Tends of Thousands of Updates/Sec
- Traverse Tens of Millions of Rows/Sec
- Comfortably Handle Several TB Database size
Lets Do Some Math

100,000 QPS
10 Queries per User Interaction
10,000 User Interactions/sec
864,000,000 User Interactions/Day
30 User Interactions/User Avg
28,000,000 Daily Active Users Possible
15M of Daily Active Users counting time of day skew
MySQL Challenges in 2018

- Increasing Security and Compliance Requirements
- Running MySQL in Containers
- Easy to Use Scale-Out Solution
- Self Tuning
- Parallel Query Execution
- GPL License
- Oracle Reputation
SAVE THE DATE!

April 23-25, 2018
Santa Clara Convention Center

OPEN SOURCE DATABASE CONFERENCE

www.perconalive.com
Thank You!