Best Practices for Migrating MySQL to the Cloud

Juan Pablo Arruti
Percona

OPEN SOURCE DATABASE CONFERENCE

PERCONA LIVE

Agenda

- IaaS vs DBaaS
- Migrating Data
- Replication between On-Premises and Cloud
- Testing Cloud Environments
- High Availability
- Monitoring
- Backups

IaaS vs DBaaS

What is IaaS?

Infrastructure as a Service (IaaS)

- Fundamentals Compute Resources
    - Servers, Storage, Network
- Provisioned and managed over the internet
- Complete control of servers
- AWS Services
    - EC2, EBS, VPC

And DBaaS?

Database as a Service (DBaaS)

- Provides database service (Instance or Cluster)
- Targeted to easily
  - Setup, Operate, Scale
- Manages common administration tasks
  - Backups, Patching, Failure Detection, Failover
- No OS access
- No Super privilege
- AWS
  - **RDS**, DynamoDB, Redshift

Pros and Cons of IaaS

Pros

- More control and flexibility
- Wide Instance Types
- Cheaper than RDS

Cons

- More operational work

Pros and Cons of DBaaS

Pros

- Easy to Manage
- Less operational tasks

Cons

- Less control and flexibility
- More expensive
- Limited Instance types

Which Do You Choose?

IaaS

- Database needs specific tuning or feature
- Available resources for operational tasks

DBaaS

- Need focus on data and code
- Generic setups are okay

Migrating Data

Best Practices

- Make it simple
- Migrate to same or higher minor version
- Avoid major version upgrades

Migrating Data to IaaS

- Similar to on-premises databases
- Use Physical Backups for large databases
  - XtraBackup, Cold Backups
- Logical Backups for small databases

Migrating Data to DBaaS

- Logical Backups
  - Access through MySQL Client
  - mysqldump, mysqlpump, mydumper

- Physical Backups are possible
  - XtraBackup*

Only available for AWS RDS

Migrating Data to DBaaS

Available MySQL Client Tools

- mysqldump
  - Most adopted tool
  - Single-threaded
- mysqlpump
  - Introduced in MySQL 5.7
  - Parallel backups
  - Restores are Single-threaded
- mydumper/myloader
  - Parallel backups and restores

Migrating Data to DBaaS

Best Practices for MySQL Clients

- Export all objects first
  - `--no-data --routines --events --triggers`
- Then export only data
  - `--no-create-info --no-create-db`
  - `--routines=no --events=no --triggers=no`
- Enable `log_bin_trust_function_creators` if `log_bin=1`
- Change object definer
  - ``DEFINER=`user`@`host` ``
- Force load and check all errors
  - `--force`

Migrating Data to RDS

Best Practices for MySQL Clients

- Increase `max_allowed_packet` (Default 4 MB)

- `time_zone` can be modified in parameter group (Default UTC)
    - RDS uses mysql schema Time Zone Tables
    - Recommended
        - Set session `time_zone` to match source database

Migrating Data to RDS

Speeding Up Logical Restore

- EC2 and RDS in same AZ
- Disable Multi-AZ
- Increase IOPS
- Modify Default Settings
  - Relax Durability
    - `sync_binlog != 1`
    - `innodb_flush_log_at_trx_commit != 1`
  - Tune InnoDB
    - Increase `innodb_log_file_size` (Default 128 MB)
    - Increase `innodb_buffer_pool_size` (Default DBInstanceClassMemory*3/4)

Restore Amazon RDS from Xtrabackup

- Overview
  - Take backup from database
  - Upload into S3 bucket
  - Create new instance from the backup
    - Amazon MySQL RDS
    - Amazon Aurora MySQL

Restore Amazon RDS from Xtrabackup

- Limitations
    - Supported MySQL 5.6 and 5.7
    - Source/Target major versions must match
        - Target minor version must be higher
    - Source tables defined within default datadir
    - 6 TB database size limit
    - Source database can't be encrypted
    - User accounts, functions, stored procedures and time zone info are not imported automatically

Replication Between On-Premises and Cloud

Replicating to the Cloud

- IaaS
  - Same as replication in on-premises
- DBaaS
  - Implementation and its limitations depends on the cloud provider

Replicating to the Cloud

Best Practices

- If latency is high
  - Use compression for Master/Slave protocol
    - `slave_compressed_protocol=1`
  - Monitor replication lag with **pt-heartbeat**

- Ensure tables have Primary Key
  - `binlog_format = ROW`

External Master on AWS RDS

- Easy to set
- NO `binlog_format` constraints (MIXED, ROW, STATEMENT)
  - Recommended ROW to avoid `time_zone` mismatch
- Log File Position or GTID based
- No filtered Replication is allowed
- Replication administration using procedures
  - `mysql.rds_set_external_master`
  - `mysql.rds_start_replication`

    ...

Testing Cloud Environments

Why Benchmark Cloud?

- Cloud resources may not map directly
- Validate if cloud instance is able to handle traffic
- Choose between IaaS and DBaaS
- Available tools
  - sysbench, pt-upgrade**, Query Playback, ProxySQL**

Query Playback

Key aspects

- Percona Labs GitHub repository
  - No active development
- Executes Queries in logs
  - Slow Query log, General log
- Compares execution results with Log
- Servers data should be consistent
- NO read-only option
- Multi-threaded
  - Queries executed at arrival time

Query Playback

Example Report

```
SELECT c FROM sbtest37 WHERE id=505;  -->
thread 67 slower query was run in  86 microseconds instead of 34

Detailed Report
----------------
SELECTs  : 1858522 queries (19297 faster, 1839225 slower)
...

Report
------
Executed 2161872 queries
Spent 00:09:08.631886 executing queries versus an expected 00:04:43.697328 time.
23610 queries were quicker than expected, 2138262 were slower
A total of 0 queries had errors.
Expected 40606531 rows, got 40606533 (a difference of 2)
Number of queries where number of rows differed: 2.

Average of 113782.74 queries per connection (19 connections).
```

ProxySQL

What is ProxySQL?

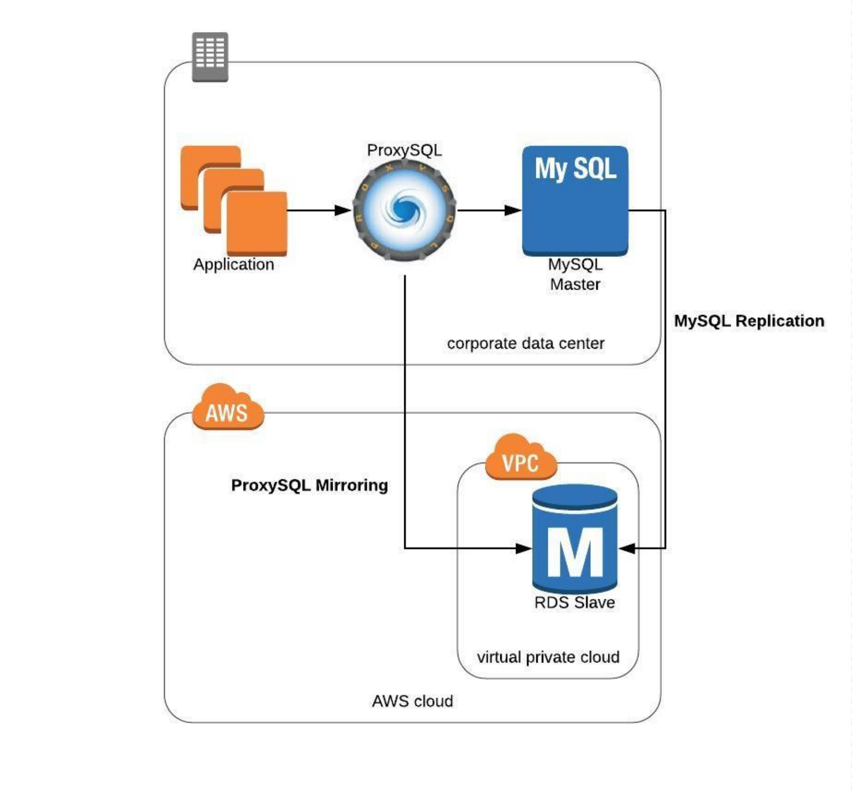
- GPL High Performance MySQL Proxy
- MySQL Protocol Aware
  - Clients connect to ProxySQL
  - Requests are evaluated
  - Actions are performed
    - Routing, Re-write, **Mirroring**

ProxySQL Query Mirroring

How does it work?

- Each client executes a query
- ProxySQL receives each query
- Query Processor identifies if the query is Mirrored
- Associates the Query to a Thread Pool
- Executes each Query in the Pool

ProxySQL Query Mirroring

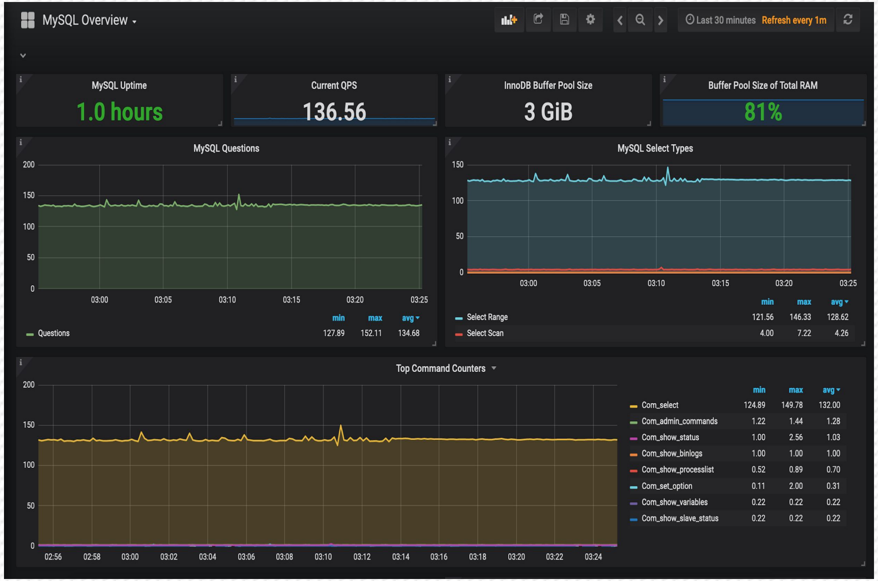


ProxySQL Query Mirroring

Limitations

- May execute Queries in different order
- Some Queries may not be executed
- Data consistency is not guaranteed
- Adds load to ProxySQL process
- Prepared statements are not supported
- No report is provided

Query Mirroring Example



MySQL Overview

Last 30 minutes Refresh every 1m

MySQL Uptime	Current QPS	InnoDB Buffer Pool Size	Buffer Pool Size of Total RAM
1.0 hours	136.56	3 GiB	81%

MySQL Questions

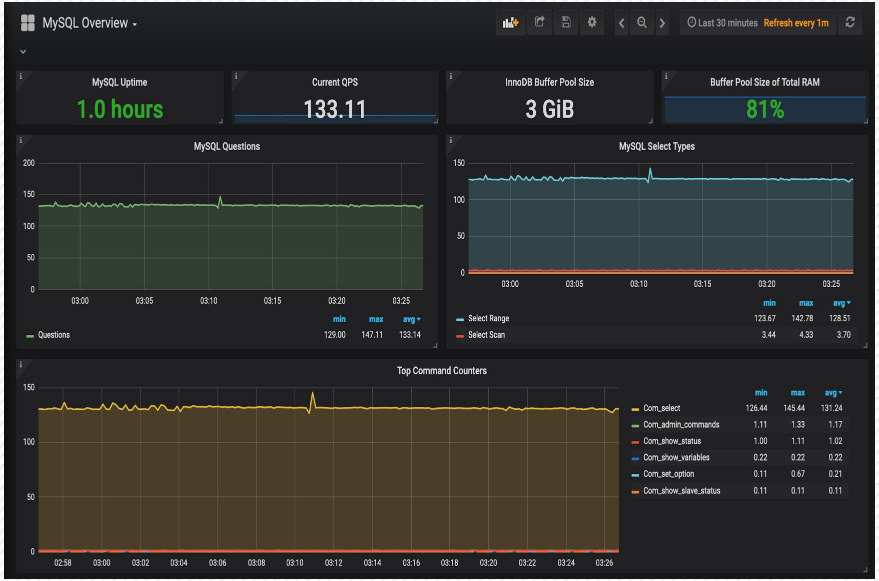
	min	max	avg
Questions	127.89	152.11	134.68

MySQL Select Types

	min	max	avg
Select Range	121.56	146.33	128.62
Select Scan	4.00	7.22	4.26

Top Command Counters

	min	max	avg
Com_select	124.89	149.78	132.00
Com_admin_commands	1.22	1.44	1.28
Com_show_status	1.00	2.56	1.03
Com_show_binlogs	1.00	1.00	1.00
Com_show_processlist	0.52	0.89	0.70
Com_set_option	0.11	2.00	0.31
Com_show_variables	0.22	0.22	0.22
Com_show_slave_status	0.22	0.22	0.22



MySQL Overview

Last 30 minutes Refresh every 1m

MySQL Uptime	Current QPS	InnoDB Buffer Pool Size	Buffer Pool Size of Total RAM
1.0 hours	133.11	3 GiB	81%

MySQL Questions

	min	max	avg
Questions	129.00	147.11	133.14

MySQL Select Types

	min	max	avg
Select Range	123.67	142.78	128.51
Select Scan	3.44	4.33	3.70

Top Command Counters

	min	max	avg
Com_select	126.44	145.44	131.24
Com_admin_commands	1.11	1.33	1.17
Com_show_status	1.00	1.11	1.02
Com_show_variables	0.22	0.22	0.22
Com_set_option	0.11	0.67	0.21
Com_show_slave_status	0.11	0.11	0.11

Query Mirroring Example

```
mysql> select * from stats_mysql_query_digest where digest_text like 'SELECT c FROM sbtest1 %' ORDER BY
hostgroup  \G
*************************** 1. row ***************************
  hostgroup: 1
 schemaname: sbtest
   username: jptest
      digest: 0x290B92FD743826DA
digest_text: SELECT c FROM sbtest1 WHERE id BETWEEN ? AND ?
 count_star: 48864
 first_seen: 1558761934
  last_seen: 1558765725
   sum_time: 14788745778
   min_time: 2877
   max_time: 3733095
*************************** 2. row ***************************
  hostgroup: 2
 schemaname: sbtest
   username: jptest
      digest: 0x290B92FD743826DA
digest_text: SELECT c FROM sbtest1 WHERE id BETWEEN ? AND ?
 count_star: 48832
 first_seen: 1558761936
  last_seen: 1558765725
   sum_time: 16477562501
   min_time: 2786
   max_time: 4651554
```

High Availability

Load Balancers

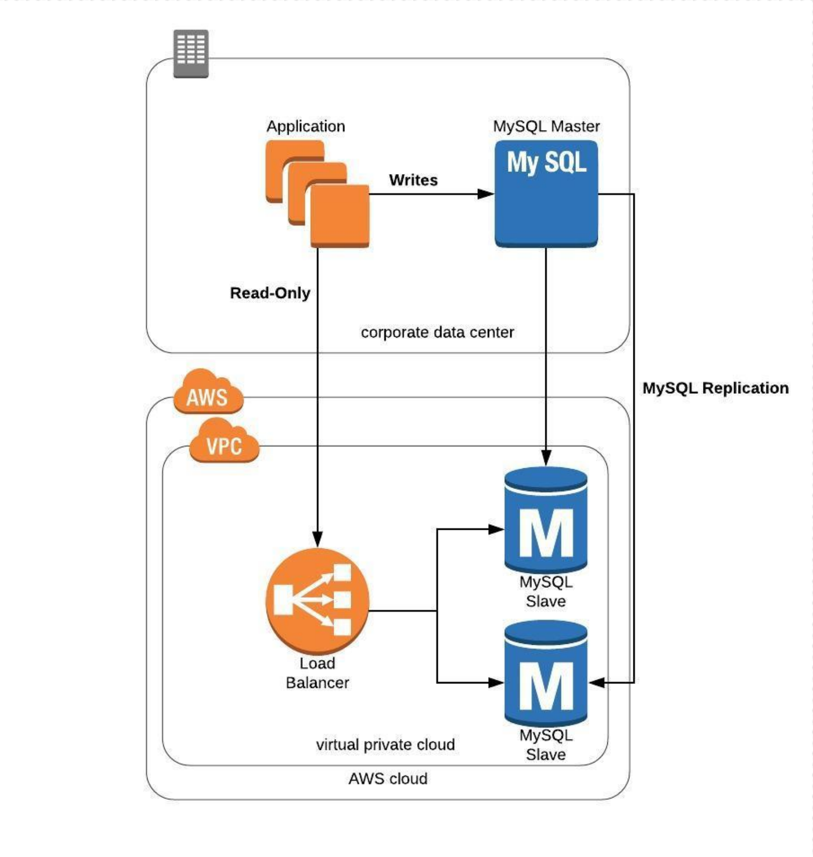
Why are they useful?

- Phased migration approach
- Routing read-only traffic
- Monitor the new environment
- Adjust it if necessary
- Confirm the environment is stable
- Minimize downtime
- Avoid modifying App connection string
- Available Load Balancers
  - **Cloud Load Balancing**
  - **ProxySQL**

Cloud Load Balancing

- TCP Load Balancer
- Distributes traffic
- AWS Elastic Load Balancer
    - RDS not supported
    - Routes traffic only across Availability Zones
    - Custom MySQL health checks can be used

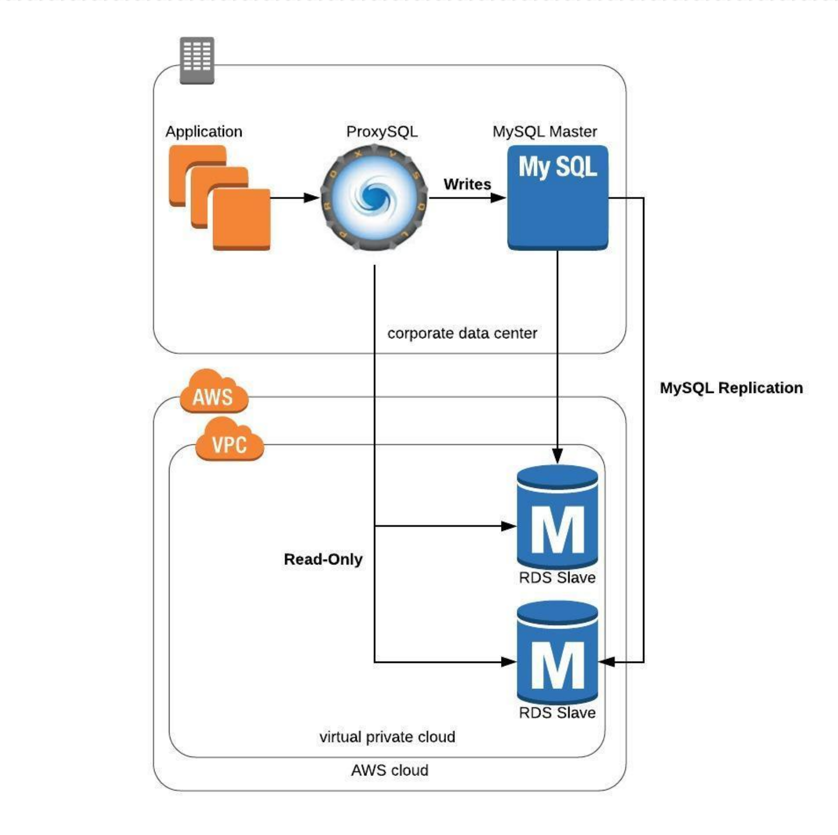
AWS ELB - Phased Migration



ProxySQL

- SQL Connection Endpoint
- Routes traffic to MySQL databases
- Splits read/write traffic
  - Based on `read_only` value
- Supports DBaaS

ProxySQL - Query Routing



DBaaS High Availability

How is it achieved?

- Data redundancy
- Automatic failover

How is it implemented?

- Each cloud vendor implements it differently

AWS RDS - Multi-AZ

Key aspects

- Synchronous Standby Replica - DRBD
- Block replicated to Different Availability Zone than Primary
- Secondary is used for backups
- Failover takes place by internal DNS change

Limitations

- Reads on Secondary are not possible
- DML Overhead

AWS RDS Read-Replicas

Key aspects

- MySQL asynchronous replication
- Scale-out Reads
- Promote to stand-alone database
- Within Same AZ, Cross AZ, Cross Region
- Easy to implement

Monitoring

Monitoring MySQL in the Cloud

Best practices

- Establish a baseline
- Measure workload under different conditions
- Compare cloud instances and on-premises

Available tools

- Cloud Monitoring System
- Percona Monitoring and Management

AWS CloudWatch

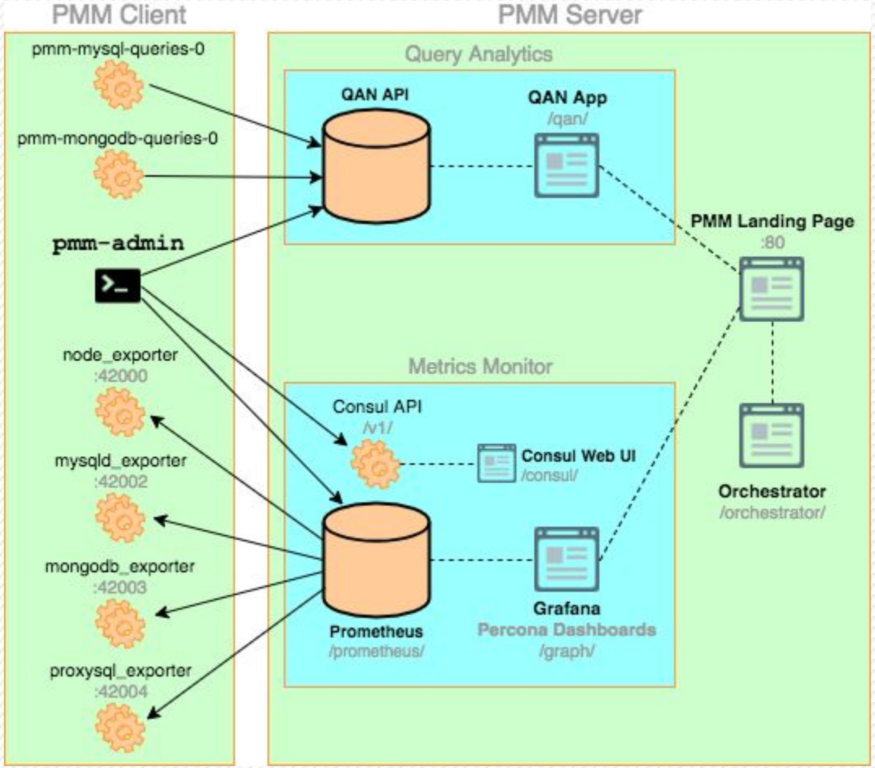
What is CloudWatch?

- Monitors AWS resources
- Metrics collected automatically
- Custom dashboards
- Create alarms and send notifications based on metrics
- Launch additional resources or stop them
- High-level Database metrics
  - `BinLogDiskUsage`
  - `DatabaseConnections`
  - `ReplicaLag`

Percona Monitoring and Management

- Open source monitoring platform
- MySQL, MongoDB, PostgreSQL and ProxySQL
- Detailed metrics for DB Servers
- Query Analytics shows current queries and stats
- Supports AWS MySQL RDS and Aurora

PMM Architecture



Monitoring RDS in PMM

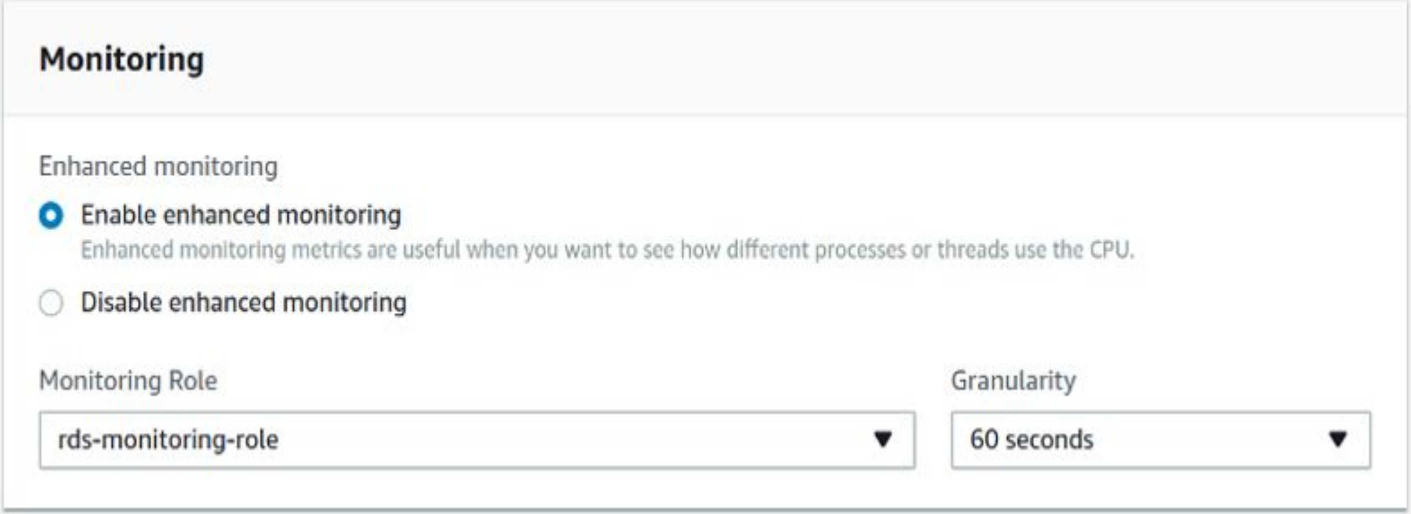
Recommendations

- PMM Server in same AZ than RDS
- Don't use T2 instances
- Use Elastic IP address
- Don't use RDS admin user

Monitoring RDS in PMM

- Enable `performance_schema` for Query Analytics

- Enable enhanced monitoring
    - OS level metrics



Monitoring

Enhanced monitoring

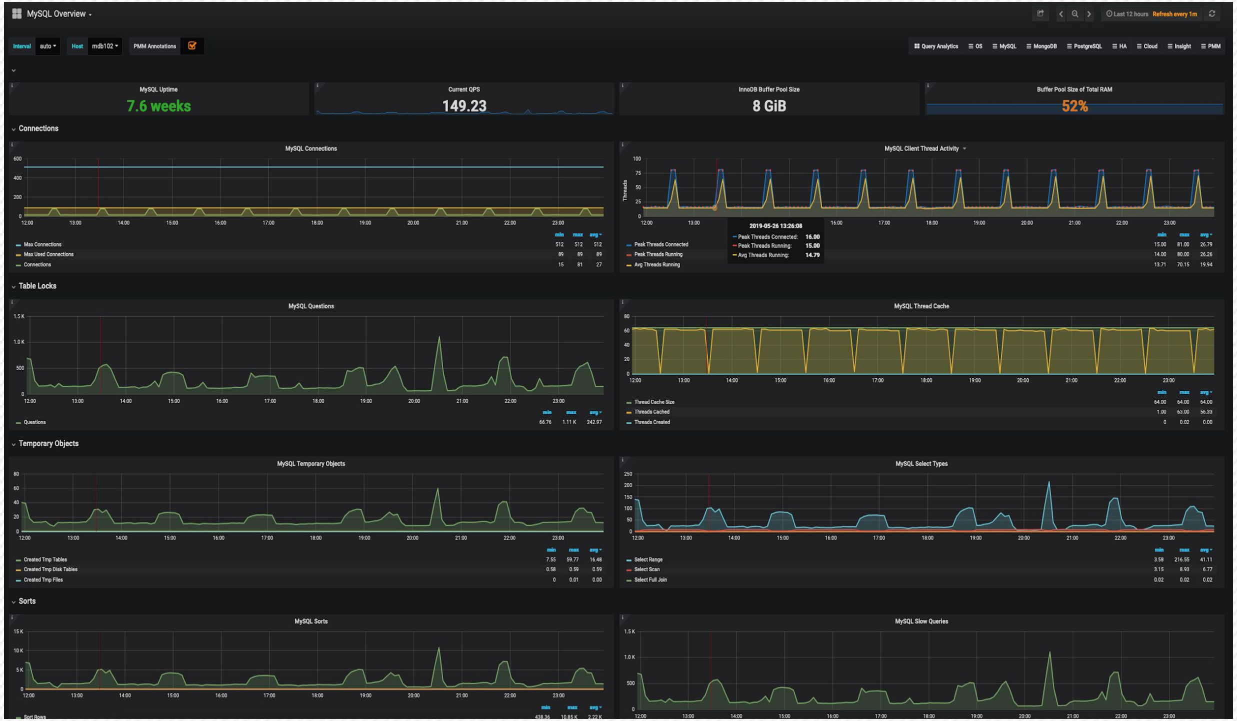
● Enable enhanced monitoring
Enhanced monitoring metrics are useful when you want to see how different processes or threads use the CPU.

○ Disable enhanced monitoring

Monitoring Role: rds-monitoring-role ▼

Granularity: 60 seconds ▼

PMM MySQL Overview Dashboard



MySQL Overview

Interval: auto | Host: mdb102 | PMM Annotations ☑

Last 12 hours Refresh every 1m

MySQL Uptime: 7.6 weeks

Current QPS: 149.23

InnoDB Buffer Pool Size: 8 GiB

Buffer Pool Size of Total RAM: 52%

Connections

MySQL Connections

	min	max	avg
Max Connections	512	512	512
Max Used Connections	89	89	89
Connections	15	81	27

MySQL Client Thread Activity

2019-05-26 13:26:08
Peak Threads Connected: 16.00
Peak Threads Running: 15.00
Avg Threads Running: 14.79

	min	max	avg
Peak Threads Connected	15.00	81.00	26.79
Peak Threads Running	14.00	80.00	26.26
Avg Threads Running	13.71	70.15	19.94

Table Locks

MySQL Questions

	min	max	avg
Questions	66.76	1.11 K	242.97

MySQL Thread Cache

	min	max	avg
Thread Cache Size	64.00	64.00	64.00
Threads Cached	1.00	63.00	56.33
Threads Created	0	0.02	0.00

Temporary Objects

MySQL Temporary Objects

	min	max	avg
Created Tmp Tables	7.55	59.77	16.48
Created Tmp Disk Tables	0.58	0.59	0.59
Created Tmp Files	0	0.01	0.00

MySQL Select Types

	min	max	avg
Select Range	3.58	216.55	41.11
Select Scan	3.15	8.93	6.77
Select Full Join	0.02	0.02	0.02

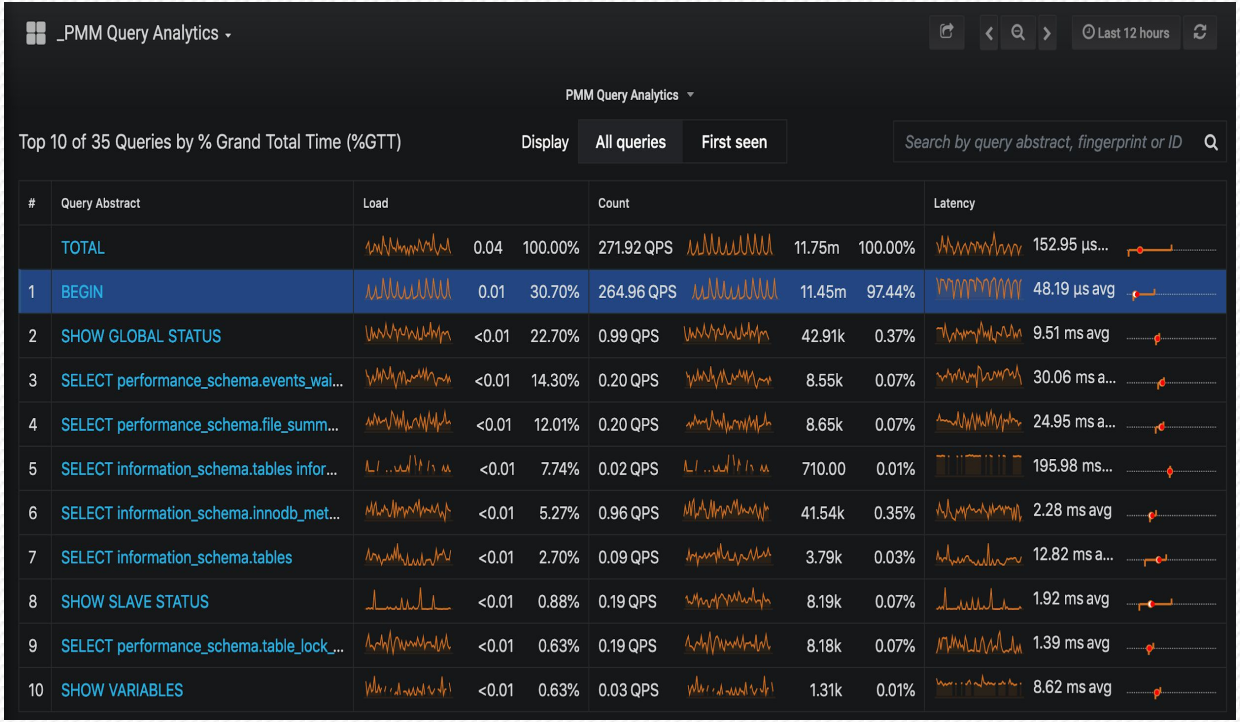
Sorts

MySQL Sorts

	min	max	avg
Sort Rows	438.36	10.85 K	2.22 K

MySQL Slow Queries

PMM Query Analytics



Top 10 of 35 Queries by % Grand Total Time (%GTT)

#	Query Abstract	Load		Count			Latency
	TOTAL	0.04	100.00%	271.92 QPS	11.75m	100.00%	152.95 µs...
1	BEGIN	0.01	30.70%	264.96 QPS	11.45m	97.44%	48.19 µs avg
2	SHOW GLOBAL STATUS	<0.01	22.70%	0.99 QPS	42.91k	0.37%	9.51 ms avg
3	SELECT performance_schema.events_wai...	<0.01	14.30%	0.20 QPS	8.55k	0.07%	30.06 ms a...
4	SELECT performance_schema.file_summ...	<0.01	12.01%	0.20 QPS	8.65k	0.07%	24.95 ms a...
5	SELECT information_schema.tables infor...	<0.01	7.74%	0.02 QPS	710.00	0.01%	195.98 ms...
6	SELECT information_schema.innodb_met...	<0.01	5.27%	0.96 QPS	41.54k	0.35%	2.28 ms avg
7	SELECT information_schema.tables	<0.01	2.70%	0.09 QPS	3.79k	0.03%	12.82 ms a...
8	SHOW SLAVE STATUS	<0.01	0.88%	0.19 QPS	8.19k	0.07%	1.92 ms avg
9	SELECT performance_schema.table_lock_...	<0.01	0.63%	0.19 QPS	8.18k	0.07%	1.39 ms avg
10	SHOW VARIABLES	<0.01	0.63%	0.03 QPS	1.31k	0.01%	8.62 ms avg

Backups

Cloud Storage

Key aspects

- Object storage - AWS S3
- Designed for durability - 99.999999999%
- Highly available
- Scalable
- Secure
- Range of storage classes
  - Standard, Infrequent Access, Glacier
  - Lifecycle policies
- Off-site storage

MySQL Backups to AWS S3

mysqldump - example

- Copy dump file with aws cli

```
shell> mysqldump --all-databases > mysql.dump.sql

shell> gzip mysql.dump.sql

shell> aws s3 cp mysql.dump.sql.gz s3://mysqlbucketprod/mysql.dump.sql.gz
upload: ./mysql.dump.sql.gz to s3://mysqlbucketprod/mysql.dump.sql.gz
```

or

```
shell> mysqldump --all-databases | gzip | aws s3 cp - \
         s3://mysqlbucketprod/mysql.dump.sql.gz
```

MySQL Backups to AWS S3

XtraBackup - example

- Stream files with **xbstream**
- Upload stream with **xbcloud**

```
shell> xtrabackup --backup --stream=xbstream --extra-lsndir=/tmp \
         --target-dir=/tmp | \
       xbcloud put --storage=s3 \
         --s3-endpoint='s3-us-west-2.amazonaws.com' \
         --s3-access-key='AKIAJ6HPUNXNZPTL2AAA' \
         --s3-secret-key='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' \
         --s3-bucket='mysqlbucketprod' \
         --parallel=10 \
         $(date +"%Y%m%d%H%M%S")-full_backup
```

RDS Backups

Automated backups

- Defined backup window
- Stored in S3
- InnoDB tables
- Elevated latency and IO freeze - except for Multi-AZ
- Point in time recovery
- Retention period = 1 - 35 days

Database snapshots

- Manually initiated
- No point in time recovery

Questions

Thank You to Our Sponsors

continuent

VividCortex

VERITAS
The truth in information.

aws

EDB
ENTERPRISEDB

MySQL™

Altinity

PingCAP

SmartStyle

facebook

ScaleGrid

Vendita

intel OPTANE™

GALERA CLUSTER

PROXYSQL
ProxySQL

aiven

Pythian
love your data®

MariaDB®
FOUNDATION

FREE SOFTWARE
FOUNDATION

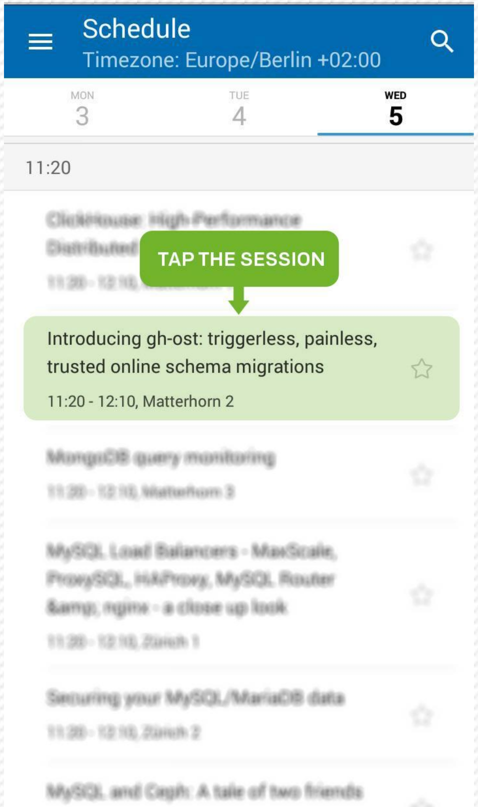
PlanetScale

Bloomberg
Engineering

yelp®

AUSTIN
TECHNOLOGY
COUNCIL

Rate My Session



ClickHouse: High Performance Distributed...

11:20 - 12:10, Matterhorn ...

TAP THE SESSION

Introducing gh-ost: triggerless, painless, trusted online schema migrations

11:20 - 12:10, Matterhorn 2

MongoDB query monitoring

11:20 - 12:10, Matterhorn 3

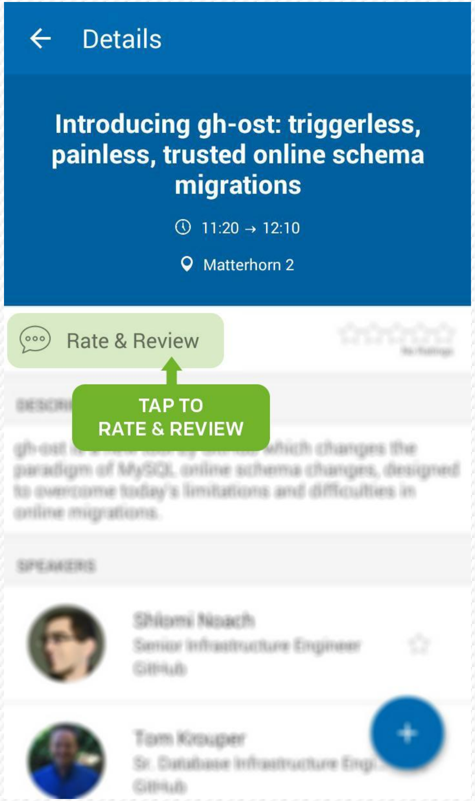
MySQL Load Balancers - MaxScale, ProxySQL, HAProxy, MySQL Router &amp; nginx - a close up look

11:20 - 12:10, Zürich 1

Securing your MySQL/MariaDB data

11:20 - 12:10, Zürich 2

MySQL and Ceph: A tale of two friends



Introducing gh-ost: triggerless, painless, trusted online schema migrations

11:20 → 12:10

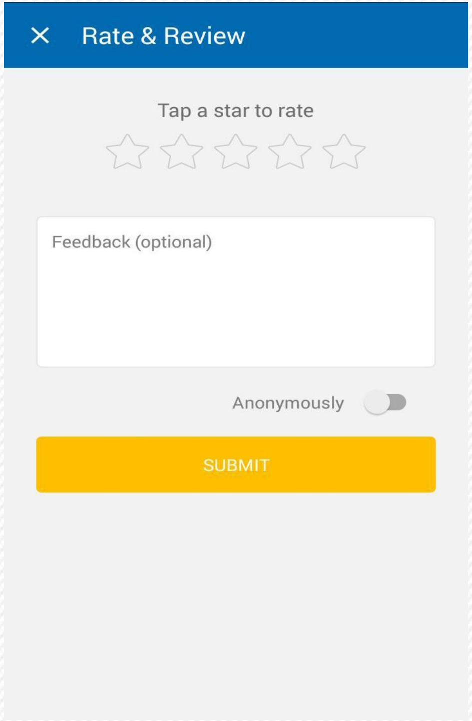
Matterhorn 2

TAP TO RATE & REVIEW

gh-ost is a new tool by GitHub which changes the paradigm of MySQL online schema changes, designed to overcome today's limitations and difficulties in online migrations.

Shlomi Noach
Senior Infrastructure Engineer
GitHub

Tom Krouper
Sr. Database Infrastructure Eng...
GitHub



Thank you!