The Care and Feeding of a MySQL Database for Linux Administrators

Dave Stokes
MySQL Community Manager
David.Stokes@Oracle.com
Simple Introduction

This is a general introduction to running a MySQL database server(s) for Linux Administrator
Simple Introduction

This is a general introduction to running a MySQL database server(s) for Linux Administrator

Database servers have needs different that SMTP, HTTP, or other servers
Simple Introduction

This is a general introduction to running a MySQL database server(s) for Linux Administrator

Database servers have needs different that SMTP, HTTP, or other servers

Hardware choices are critical! (do not go cheap)
Simple Introduction

This is a general introduction to running a MySQL database server(s) for Linux Administrator

Database servers have needs different that SMTP, HTTP, or other servers

Hardware choices are critical! (do not go cheap)

Tuning to 80% efficiency is relatively easy
Simple Introduction

This is a general introduction to running a MySQL database server(s) for Linux Administrator

Database servers have needs different that SMTP, HTTP, or other servers

Hardware choices are critical! (do not go cheap)

Tuning to 80% efficiency is relatively easy (last 20% is tricky)
Session Overview

1. Basics of a database server
2. Hardware
3. MySQL Configuration
4. Monitoring Operations
5. Backups
6. Replication
7. Indexes
How does a Database server work

Client

SELECT phone
FROM friends
WHERE name = 'Joe';

Server
Who does a Database server work

Client

SELECT phone
FROM friends
WHERE name = 'Joe';

Server

PARSE

find Joe in friends table in memory
return phone
Who does a Database server work

Client

SELECT phone
FROM friends
WHERE name = 'Joe';

Server

PARSE
find Joe in friends table in memory
return phone

Process phone data
Who does a Database server work

Client

SELECT phone
FROM friends
WHERE name = 'Joe';

Server

PARSE

find Joe in friends table in memory

return phone

What was that about memory???
Rule #1

- Databases love data in memory
Rule #1

- Databases love data in memory

Corollary #1 – getting data in/out of memory will cause you nightmares!
What if it is not in memory?

MySQL

Please give me the data from the city table

OS
What if it is not in memory?

MySQL

Please give me the

OS

Get inode
data from the city
table
What if it is not in memory?

MySQL
Please give me the data from the city table

OS
Get inode
Ask disk for data
What if it is not in memory?

MySQL

Please give me the data from the city table

OS

Get inode

Ask disk for data

Get data into buffer
What if it is not in memory?

MySQL
Please give me the data from the city table

OS
Get inode
Ask disk for data
Get data into buffer
Hand buffer off

Load data into memory
What if it is not in memory?

MySQL

Please give me the data from the city table

OS

Get inode

Ask disk for data

Get data into buffer

Hand buffer off

Much longer than just reading from memory

Load data into memory
Rule #2

• Databases have to do unpredictable queries, random I/O, and sequential scans so slow I/O kills performance
Rule #2

- Databases have to do unpredictable queries, random I/O, and sequential scans so slow I/O kills performance

Corollary #2 – You need to have good gear

   or

   going cheap = going slow
Hardware recommendations

1. Memory – lots of it, ecc
Hardware recommendations

1. Memory – lots of it, ecc
2. DISKs – more spindles, high speed, fast controllers, RAID 10, write back cache, and XFS/ZFS not ext2/3
Hardware recommendations

1. Memory – lots of it, ecc

2. DISKs – more spindles, high speed, fast controllers, RAID 10, write back cache, and XFS/ZFS not ext2/3

3. Write-through caches with battery backup units for disks must be monitored, and have life span much longer than planned outages
Hardware recommendations

1. Memory – lots of it, ecc

2. DISKs – more spindles, high speed, fast controllers, RAID 10, write back cache, and XFS/ZFS not ext2/3

3. Write-through caches with battery backup units for disks must be monitored, and have life span much longer than planned outages

4. CPUs, Core less important
Installation

1. Use prebuilt packages
Installation

1. Use prebuilt packages
2. Don’t double up with other services
Installation

1. Use prebuilt packages
2. Don’t double up with other services
3. Supplied configuration files are OLD!
Installation

1. Use prebuilt packages
2. Don’t double up with other services
3. Supplied configuration files are OLD!
4. Move logs to different disk than data
Installation

1. Use prebuilt packages
2. Don’t double up with other services
3. Supplied configuration files are OLD!
4. Move logs to different disk than data
5. Spread data over different drives
Installation

1. Use prebuilt packages
2. Don’t double up with other services
3. Supplied configuration files are OLD!
4. Move logs to different disk than data
5. Spread data over different drives
6. Backups are necessary – and practice recovery!
Monitoring Operations

1. Slow query log -- not all long running queries are bad
Monitoring Operations

1. Slow query log -- not all long running queries are bad
2. Log queries not using indexes
Monitoring Operations

1. Slow query log -- not all long running queries are bad
2. Log queries not using indexes
3. Use monitoring software – MEM, phpMyAdmin, Nagios, etc – and pay attention to it
Monitoring Operations

1. Slow query log -- not all long running queries are bad
2. Log queries not using indexes
3. Use monitoring software – MEM, phpMyAdmin, Nagios, etc – and pay attention to it
4. More in tuning ...
Backups

Backups are usually some sort of disk snapshot or serializing data to a file.
Backups

Backups are usually some sort of disk snapshot or serializing data to a file.

The more the better but you need to know steps to recover dropped table, lost databases, or mangled data.
Backups

Backups are usually some sort of disk snapshot or serializing data to a file.

The more the better but you need to know steps to recover dropped table, lost databases, or mangled data.

Use data replication to a slave and then backup slave.
Backups

Backups are usually some sort of disk snapshot or serializing data to a file.

The more the better but you need to know steps to recover dropped table, lost databases, or mangled data.

Use data replication to a slave and then backup slave.

Be paranoid!!!!!!
Replication

Replication for MySQL is the binary log for the master being copied to a slave. The slave then updates its copy of the data.
Replication for MySQL is the binary log for the master being copied to a slave. The slave then updates its copy of the data

Two types:

1. Asynchronous – server does not check changes sent to slave before proceeding
Replication

Replication for MySQL is the binary log for the master being copied to a slave. The slave then updates its copy of the data.

Two types:

1. Asynchronous – server does not check changes sent to slave before proceeding
2. Semi Synchronous – server checks that server received changes before proceeding
Replication -- threads

Currently single threaded – 5.6 will fix that
Replication -- network

Network latency will effect MySQL replication. So plan network topology to minimize bandwidth competition with other systems/services.
Replication -- network

Network latency will effect MySQL replication. So plan network topology to minimize bandwidth competition with other systems/services.

Slaves do not need to be as fast as the master but try to keep things reasonably close
Replication -- network

Network latency will effect MySQL replication. So plan network topology to minimize bandwidth competition with other systems/services.

Slaves do not need to be as fast as the master but try to keep things reasonably close

Do not have to replicate all tables/databases to all slaves. Cut down on traffic by replicating what is needed!
Indexes are good

Without Index

DB needs to scan entire table or table scan

With Index

DB can go right to record
Indexes, the bad

- Overhead -- space, speed, maintenance
Indexes, the bad

- Overhead -- space, speed, maintenance
- Not a panacea – does not cure all problems
Indexes, the bad

- Overhead -- space, speed, maintenance
- Not a panacea – does not cure all problems
- Will not help if you need to perform a table scan
Indexes, the bad

• Overhead -- space, speed, maintenance
• Not a panacea – does not cure all problems
• Will not help if you need to perform a table scan
• Composite indexes can be tricky – YearMonthDay usually better than DayMonthYear
Tuning to 80%

- Use InnoDB
Tuning to 80%

- Use InnoDB
- Set `innodb_buffer_pool_size` to 70-80% of memory
Tuning to 80%

- Use InnoDB
- Set `innodb_buffer_pool_size` to 70-80% of memory
Tuning to 80%

- Use InnoDB
- Set `innodb_buffer_pool_size` to 70-80% of memory
- Use XFS
Tuning to 80%

- Use InnoDB
- Set `innodb_buffer_pool_size` to 70-80% of memory
- Use XFS
- Partition data -- divide and conquer
Tuning to 80%

- Use InnoDB
- Set `innodb_buffer_pool_size` to 70-80% of memory
- Use XFS
- Partition data -- divide and conquer
- Architect your data
Tuning to 80%

- Use InnoDB
- Set `innodb_buffer_pool_size` to 70-80% of memory
- Use XFS
- Partition data -- divide and conquer
- Architect your data
- Review your SQL statements
Tuning Past 80%
Q&A