

ORACLE®



**ORACLE®**

## **MySQL Server Performance Tuning 101**

Ligaya Turmelle  
Senior Technical Support Engineer - MySQL  
ligaya.turmelle@oracle.com  
<http://joind.in/2285>

# MySQL

- world's most popular open source database software
- a key part of LAMP (Linux, Apache, MySQL, PHP / Perl / Python)
- Site: <http://www.mysql.com/>
- Download: [mysql.org](http://mysql.org) or <http://dev.mysql.com/downloads/>
- Online Manual: <http://dev.mysql.com/doc/refman/5.1/en/index.html>

**Before we start**



ORACLE

# Before you Start

- Single **greatest** gain can be received by optimizing the queries
- Optimize the underlying system
- Understand that there are no hard/fast/exact answers for a “proper” value
  - Change one value at a time and benchmark to look for improvement
  - Better to under allocate then to over allocate

# How MySQL Uses Memory

- 2 ways - Global and Per Connection
- Global - Most allocated once when the server starts - can be large
- Per Connection - for \*each\* connection as needed - should be small
- Global memory + (max connections \* session buffers)

# Information

- Current Settings

- mysql> SHOW GLOBAL VARIABLES;
- Shows the majority of the settings

```
+-----+
|Variable_name          | Value
+-----+
| auto_increment_increment | 1
| auto_increment_offset   | 1
| automatic_sp_privileges | ON
| back_log                | 50
| basedir                 | /Users/lig/mysql_install/mysql-enterprise-....
| binlog_cache_size       | 32768
| bulk_insert_buffer_size | 8388608
| character_set_client    | latin1
```

- my.cnf or my.ini

## Other Information

- How much RAM is on the machine?
- Is the machine dedicated to MySQL?
- 64 bit processor?
- 64 bit MySQL server binary?

# Server Status

- `mysql> SHOW GLOBAL STATUS;`
- has various counters
- gives you a feel of what the server is actually doing - not what you \*think\* it is doing
- issue the command twice with time between
  - this allows you to calculate the changes in values (Delta) over time

# Example Status

Variable_name	Value
Aborted_clients	6618
Aborted_connects	3957
Binlog_cache_disk_use	0
Binlog_cache_use	0
Bytes_received	1320510015
Bytes_sent	2978960756
Com_admin_commands	32124
....	
Threads_cached	0
Threads_connected	1
Threads_created	316771
Threads_running	1
Uptime	1722523
Uptime_since_flush_status	1722523

# Finding the Delta

- Uptime - part of the Status information and is given in seconds
  - Ex: 1086122 seconds = ~19.9 days = 478.5 hrs
- helps you calculate the values in a given time frame (high load, average load)
  - Ex:  $(\text{bytes received}[2] - \text{bytes received}[1]) / (\text{Uptime}[2] - \text{Uptime}[1])$   
 $(1320582719 - 1320510015) / (1722729 - 1722523) =$   
 $72704 \text{ bytes} / 206 \text{ sec} = 352.9 \text{ bytes/sec}$

**Lets** begin



ORACLE

# What is the server doing?

- Com\_XXXX

- Counter for the number of times XXXX has executed
- One status variable for each statement

Com_delete	78813
Com_insert	100357
Com_replace	3130
Com_select	984292
Com_show_tables	459
Com_show_triggers	898
Com_create_table	1349
Com_update	285105

- Used with Uptime to find XXXX/second (INSERT/sec or DELETE/sec)

# MyISAM



ORACLE®

# What is the server doing?

(con't)

- key buffer (MyISAM)
  - Key\_blocks\_unused:
    - number of unused blocks
    - $\text{key\_buffer\_size} - \text{Key\_blocks\_unused} = \text{amount in use}$
  - Key\_blocks\_used:
    - a high-water mark that indicates the maximum number of blocks that have ever been in use at one time

# What is the server doing?

(con't)

- key buffer (MyISAM)
  - Key\_read\_requests:
    - number of requests to read a key block from the cache
    - a cache hit
  - Key\_reads:
    - number of physical reads of a key block from disk
    - a cache miss
    - large value == key\_buffer\_size is probably too small
  - cache miss rate:  
$$\text{Key\_reads} / \text{Key\_read\_requests}$$
  - Key buffer efficiency:  
1 - cache miss rate

## key\_buffer\_size

- Global variable
- the buffer used for MyISAM **index blocks**
- index blocks are buffered and shared by all threads
- the maximum allowable setting
  - 4GB on 32 bit system
  - after 5.0.52 on 64 bit systems = 4GB+
- dedicated machine - 25% of total memory is common
- uses the OS file cache system to cache the files
- too high == paging == bad

# What is the server doing?

(con't)

- MyISAM table locks
  - Table\_locks\_immediate:
    - # of table locks granted immediately to MyISAM tables
  - Table\_locks\_waited:
    - # of times we have had to wait to get a table lock for MyISAM tables
    - If value is high - consider changing storage engines

# InnoDB



ORACLE®

# What is the server doing?

(con't)

- Innodb\_\*
  - Has various information about the InnoDB tablespace
    - Innodb\_buffer\_pool\_pages\_data:
      - # of pages containing data (dirty or clean)
    - Innodb\_buffer\_pool\_pages\_free:
      - # of free pages
    - Innodb\_buffer\_pool\_pages\_total:
      - total size of the buffer pool, in pages
    - Innodb\_buffer\_pool\_wait\_free:
      - count of # of times we waited for pages to be flushed
      - should be a small value

## innodb\_buffer\_pool\_size

- Global variable
- size in bytes
- used to cache data and indexes of InnoDB tables (clustered indexes - remember)
- the larger the value - the less disk IO is needed to access the data
- dedicated database server - up to 80% of the physical memory
- buffers everything itself (O\_DIRECT)
- DO NOT set to large - will cause paging/swap == bad

## innodb\_log\_file\_size

- Global variable
- size in bytes of **each** log file in a log group
- balancing act
  - larger the value, the less checkpoint flush activity - less IO
  - larger the log files - longer the crash recovery time\*
- sensible values range from 1MB to 1/N-th of the buffer pool size
  - N is the number of log files in the group
- Combined size of log files must be less than 4GB

# Query Cache



# What is the server doing?

(con't)

- Query Cache Information
  - Qcache\_total\_blocks:
    - total # of blocks in the query cache
  - Qcache\_free\_blocks:
    - # of free memory blocks. Can indicate a problem with fragmentation
  - Qcache\_hits:
    - # of query cache hits
  - Qcache\_inserts:
    - # of queries added to the query cache

# What is the server doing?

(con't)

- Query Cache Information
  - Qcache\_not\_cached:
    - # of non-cached queries
  - Qcache\_queries\_in\_cache:
    - # of queries registered in the query cache
  - Qcache\_lowmem\_prunes:
    - # of queries that were deleted from the query cache because of low memory

# What is the server doing?

(con't)

- Turning Query Cache counters into information
  - Query Cache Hit Rate:
    - quick and easy way to see if you benefit from using the QC
    - QC Hit Rate =  $\text{Qcache\_hits} / (\text{Qcache\_hits} + \text{Com\_select})$
    - higher the value the better
  - check to see how often are you invalidating queries in the cache
    - Qcache\_inserts vs Com-select
    - want  $\text{Qcache\_inserts} \ll \text{Com\_select}$
    - bigger the difference - the better
- Note: keep in mind warming up the cache

## query\_cache\_size

- Global variable
- the amount of memory allocated for caching query results
- default value is 0, which disables the query cache
- allowable values are multiples of 1024
  - Other values are rounded down to the nearest multiple

## query\_cache\_type

- Global variable - but can be set at the session level
- 3 options:
  - 0 (Off) - Don't cache results in or retrieve results from the query cache
  - 1 (On) - Cache all cacheable query results except - SELECT SQL\_NO\_CACHE. Default
  - 2 (Demand) - Cache results only for cacheable queries that begin with SELECT SQL\_CACHE
- Note that the query\_cache\_size of memory is allocated even if the query\_cache\_type is set to 0

# Threads



ORACLE®

# What is the server doing?

(con't)

- Threads
  - Threads\_cached:
    - # of threads in the thread cache
  - Threads\_connected:
    - # of currently open connections
  - Threads\_created:
    - # of threads created to handle connections
      - If value is “big” - consider increasing thread\_cache\_size
      - Thread cache miss rate:
        - $\text{Threads\_created} / \text{Connections}$

## thread\_cache\_size

- Global variable but grows as needed
- number of threads to be cached for reuse
  - When clients disconnect, the threads are put in the cache until it is full
  - Requests for threads are satisfied by reusing the threads in the cache. Only when empty is a new thread created
- increase to improve performance if you have a lot of new connections
  - May not provide a notable performance increase if you already use a good thread implementation
- thread cache efficiency
$$100 - ((\text{Threads\_created} / \text{Connections}) * 100)$$

# What is the server doing?

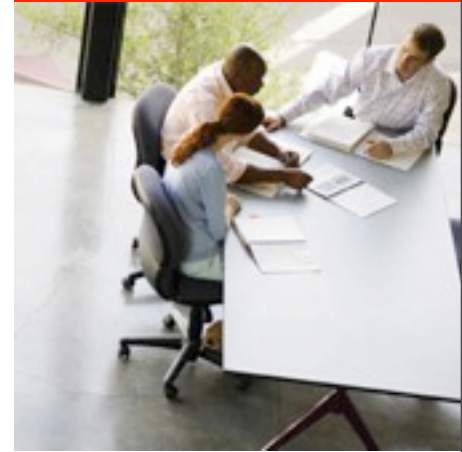
(con't)

- General Information counters of importance
  - Connections:
    - # of connection attempts (successful or not)
  - Queries:
    - # of statements executed (including within Stored Proc)
  - Questions:
    - # of statements sent to the server by clients and executed
  - Slow\_queries
    - # of queries that took longer then long\_query\_time sec.
  - Sort\_merge\_passes
    - # of merge passes that the sort algorithm had to do
      - If large - consider increasing sort\_buffer\_size

# Final Global Variable to tweak

- `table_cache`
  - Global variable that expands as needed
  - The number of open tables for all threads
    - Increasing this value increased the number of file descriptors that mysqld requires
  - If `Opened_tables` is constantly increasing and you do not use `FLUSH TABLES` often - increase this variable

# Session Variables



# Per connection variables available for tweaking

- `max_heap_table_size`
  - Dynamic variable - can be set per session
  - sets the maximum size that MEMORY tables are allowed to grow to
- `tmp_table_size`
  - Dynamic variable - can be set per session
  - maximum size of internal in-memory temporary tables
    - actual limit is determined as the smaller value for `max_heap_table_size` or `tmp_table_size`
  - If in-memory temporary table exceeds this limit, it is automatically converted to an on-disk MyISAM table.

# Per connection variables available for tweaking

(con't)

- `read_buffer_size`:
  - dynamic variable - can be set per session
  - Each thread that does a sequential scan - allocates a buffer of this size (in bytes) for each **table** it scans
  - If you do many sequential scans, consider increasing this value
  - value should be multiples of 4KB
  - maximum allowable setting is 2GB
    - Do not normally recommend it being higher than 8M

# Per connection variables available for tweaking

(con't)

- `read_rnd_buffer_size`:
  - Dynamic variable - can be set per session
  - for reading rows in sorted order following a key-sorting operation. Designed to avoid disk seeks
  - a large value can improve ORDER BY performance
  - maximum allowable setting is 2GB
    - Do not normally recommend it higher than 8M

# Per connection variables available for tweaking

(con't)

- `sort_buffer_size`:
  - dynamic variable - can be set per session
  - each thread that needs to do a sort allocates a buffer of this size
  - increase this value for faster ORDER BY or GROUP BY operations

# Per connection variables available for tweaking

(con't)

- `bulk_insert_buffer_size`:
  - Dynamic variable - can be set per session
  - cache to make bulk inserts faster for `INSERT... SELECT`, `INSERT... VALUES (...), (...), ...`, and `LOAD DATA INFILE` when adding data to no-empty tables.
  - limits the size of the cache tree in bytes per thread
  - Set to 0 disables the optimization used

# Per connection variables available for tweaking

(con't)

- `join_buffer_size`:
  - Dynamic variable - can be set per session
  - Size of the buffer that is used for plain index scans, range index scans and joins that do not use indexes - full table scans
  - The best way to get fast joins is to use indexes
  - band-aid - don't depend on it!

Questions?



ORACLE®

# Contact:

Email:

[ligaya@mysql.com](mailto:ligaya@mysql.com)

[ligaya.turmelle@oracle.com](mailto:ligaya.turmelle@oracle.com)

Personal Website:

<http://blog.khankennels.com>

Twitter: @lig

<http://joind.in/2285>



ORACLE

ORACLE®