



MySQL Server Settings Tuning

Peter Zaitsev, co-founder, Percona Ltd

April 23-26 2007

Presented by



O'REILLY

About Speaker

- Peter Zaitsev
- Co-Founder at Percona Ltd,
 - Consulting company specializing in MySQL and LAMP Performance
- Founder <http://www.mysqlperformanceblog.com>
- Spent 4.5 years with MySQL AB,
 - High Performance Group, Manager
 - Performance related Support, Development, Consulting etc
- Have designed and implemented numerous MySQL based projects

Presented by



O'REILLY

About Presentation

- What should you consider while tuning MySQL Server settings ?
- How much improvement to inspect ?
- What things to look at while setting various settings ?
- Assume you're familiar with MySQL at some level

Presented by



O'REILLY

Things to consider while tuning

- Settings are workload specific
 - There is no such thing as optimal config for 16GB of memory
- Storage Engine choice or mix is important
- Few settings depend on your OS and Hardware

Approaching tuning

- You just need to get few of them right for decent performance
 - Rest are very specific and give hard to notice improvements
- Some settings affect MySQL Behavior and data safety, be careful
- Many settings can be set on connection basics, use it.
- Optimize queries first or you may need to revisit your settings.

Typical mistakes

- Running in MySQL default settings
 - Unless you're running it on laptop to store your DVD collection data
- Using someone else's my.cnf file without checking if it matches your needs.
- Setting large global values if only one/two queries need them. Instead you can do:

```
SET sort_buffer_size=128000000;  
<run query>  
SET sort_buffer_size=DEFAULT
```

What information do we have available ?

- SHOW STATUS
- SHOW INNODB STATUS
- Operating System monitoring commands
 - vmstat
 - iostat
 - mpstat

Allocating memory

- Allocating more memory to MySQL Server often improves performance
- But Allocating too much memory may kill performance and stability
- Watch out for
 - Swapping
 - Look for swap IO rather than simple used swap space
 - Running out of address space on 32bit systems
 - Make sure OS and MySQL Server are both 64bit

Fighting Swapping

- Some OS would swap even when enough memory is available
 - Problems balancing IO cache vs Swapping
- **--memlock** – lock MySQL Server in memory
 - can be dangerous
- Tune VM subsystem to use swap only as last resort
 - echo 1 > /proc/sys/vm/swappiness
- If using Innodb on Linux try using direct IO
 - innodb_flush_method=O_DIRECT**

Know the unit in which variable is set

- **table_cache=128M**
Wrong – table cache is measured in entries
- **key_buffer_size=1024**
Wrong again, Key buffer should be specified in bytes
- **innodb_max_dirty_pages_pct=8G**
This one is set in percents

Know locality and allocation pattern

- **key_buffer_size**
Global, shared by all threads
- **sort_buffer_size**
Allocated for the thread once sorting needs to be done
- **read_buffer_size**
Allocated all at full size, even if smaller is enough
Overly large values may slow things down
larger allocations typically done from OS
- **tmp_table_size**
Specifies maximum size, will grow to this size as needed
no performance penalty for setting high.

Lets get to details

Presented by



O'REILLY

Looking at **SHOW STATUS**

- **SHOW GLOBAL STATUS** shows global server status – good for understanding the load
 - The only one available in MySQL 4.1 and below
- **SHOW LOCAL STATUS** is great for query/transaction profiling
 - Available since MySQL 5.0
 - SHOW STATUS now defaults to this
 - Some variables are global only
 - Will still be shown in SHOW STATUS output
- **mysqladmin extended -i100 -r**
 - Great to sample what is happening with MySQL Server Now

SHOW STATUS

- **Aborted_clients**

Usually no need to worry – many programs do not close connection properly

- **Aborted_connects**

May mean authentication failure, network timeout or other error

Worth to look at as may cause host blocked out

--max_connect_errors=100000

Beware password break attempts

- **Binlog_cache_disk_use/Binlog_cache_use**

how frequently bin log cache size has to spill to the disk

increase **--binlog_cache_size** if a lot

Com_XXX

- Understanding server load in terms of queries
- Queries can vary in complexity quite a lot
- Good to catch use mistakes

Unintended LOCK TABLES with Innodb tables

Com_lock_tables

Too many ROLLBACKs

Com_rollback

Prepared Statements used only once

Com_stmt_prepare = Com_stmt_execute

Temporary Objects

- **Created_tmp_tables**

Temporary tables can often be avoided by query optimization

- **Created_tmp_disk_tables**

Not enough memory is allocated – increase **tmp_table_size** and **max_heap_table_size**

Temporary table can't be created in memory

Typically because of BLOB/TEXT columns required

Can be placed on tmpfs for better performance

--tmpdir=/dev/shm on Linux

Watch out for overflow

Handler_XXX

- Represent what is happening with server on storage engine layer (better)
- Does not make a difference between different rows and different engines
- No way to see covering index usage
- **Handler_read_key, Handler_read_next** - indexed accesses
- **Handler_read_rnd, Handler_read_rnd_next** - full table scans
 - Temporary tables also counted here

Innodb_XXX

- A lot of Innodb status variables added in MySQL 5.0
- Mostly copied from **SHOW INNODB STATUS**
- Not made per-thread counters yet
- **Innodb_data_reads, Innodb_data_writes**
Physical IO to Innodb tablespace files
Use to tune **innodb_buffer_pool_size**
- **Innodb_buffer_pool_pages_misc** - how much memory Innodb locks, adaptive hash indexes, insert buffer take
- **Innodb_log_waits**
Waits happen when **innodb_log_buffer_size** is not large enough

InnoDB_XXX

- InnoDB row operations (**InnoDB_rows_read** etc) can be used to view InnoDB activity on row level
- **InnoDB_row_lock_XXX** - information about innodb lock waits and timing.
- **InnoDB_data_pending_XXX,**
InnoDB_os_log_pending_XXX
Pending IO operations. IO is overloaded if they stay high

Key Cache Info

- **Key_blocks_used / Key_blocks_unused**

How much of your key_buffer is used ?

key_blocks_used should be key_blocks_ever_used

Helps not to set key buffer higher than needed

- **Key_read_requests, Key_reads, Key_write_requests, Key_writes**

How good is your key buffer usage ?

Look at amount of reads/writes which pass to IO subsystem

Key Cache hit ratio alone is useless - checking how it is affected by buffer sizing is helpful

As hit ratio does not change much any more it may be no point increasing it further

Connections and Tables

- **Max_used_connections**

If matches **max_connections** you may have run out of connections

- **Open_files** - check not to run out of limit

- **Open_tables** – how table cache is used

- **Opened_tables** -check how quickly it grows

Grows due to explicit temporary tables

Grows if it is not large enough.

Adjust **--table-cache** variable

Make sure **--open-file-limit** is large enough

Query Cache Status

- **Qcache_hits**
How frequently query was used from query cache
Com_selects is not incremented in this case
- **Qcache_inserts**
Query stored in query cache – misses and overhead
- **Qcache_free_memory**
Free/Unused memory in query cache
Often query cache can use limited memory because of invalidation
- **Qcache_lowmem_prunes**
Not enough memory or too fragmented – remove some queries

Select Information

- **Select_full_join**
Joins without indexes. Bad and dangerous
- **Select_range**
Range scan queries. No problem with these
- **Select_range_check**
Usually queries worth looking to optimize. No good index.
- **Select_scan**
Full Table Scan. Small or Large

Sorting

- **Sort_merge_passes**
Consider increasing **sort_buffer_size** if it is high
- **Sort_range**
Sorting of ranges
- **Sort_scan**
Sorting full table scans
- **Sort_rows**
How many rows sorted per second

Table locks information

- **Table_locks_immediate**
Table locks granted without waiting
- **Table_locks_waited**
Table locks which had to be waited for
- Helpful to find out if Table locks are problem
- Do not show how long wait was for
Rare but long table locks may be too bad already

Threads Information

- **Threads_cached**

entries currently used in threads_cache

- **Threads_connected**

current number of connected threads. Make sure you have some gap to max_connections

- **Threads_created**

threads_cache “misses” increase threads_cache if it is high

- **Threads_running**

Amount of threads running at the same time Keep it reasonable

Waiting on table locks, row level locks, IO is also counted as running

SHOW INNODB STATUS - Mutexes

- InnoDB Mutexes information
- Helpful to tune **--innodb_thread_concurrency**
If number of OS Waits or Spinlocks is high it is often worth to increase it

What is “High” ?

More than 1000 OS Waits/sec

More than 100.000 spin rounds/sec

```
OS WAIT ARRAY INFO: reservation count 419870, signal count 418867
Mutex spin waits 1110481, rounds 3157468, OS waits 74647
RW -shared spins 650429, OS waits 320432; RW -excl spins 44900, OS waits 20070
```

Presented by



O'REILLY

Purge activity and memory usage

- “Total memory allocated 13867961114; in additional pool allocated 1048576”
 - Tune additional pool size
 - Check how much memory Innodb really allocated
 - May be more than you think due to Innodbs own table cache
- “Trx id counter 0 458146886 Purge done for trx's n:o < 0 458146877 undo n:o < 0 History list length 31 Total number of lock structs in row lock hash table 0”
 - Check check how purge is happening
 - set **innodb_max_purge_lag** if it is unable to keep up
 - Will slow updates down

Most important settings to set

Presented by



O'REILLY

Memory Settings - Global

- **key_buffer_size**

Used by MyISAM tables to cache Index only, not data

30% of memory for MyISAM only system

4GB limit per key buffer, can have multiple of these

MyISAM tables used for temporary tables anyway

- **innodb_buffer_pool_size**

Used by InnoDB tables

70% of memory for InnoDB only system

InnoDB performance is very critical to this setting

- **query_cache_size**

Set if using query cache

Memory Settings - Local

- **read_buffer_size, read_rnd_buffer_size**
sequential read buffers used by MyISAM and some others
Allocated when needed.
1M/4M typically good to start
- **sort_buffer_size**
buffer used for sorting. Increase if a lot of large sorts are done
- **tmp_table_size**
Maximum size of in-memory table allowed (used by complex queries)
May need more than one temporary table per query
max_heap_table_size limit is also used

Other caches

- **table_cache**

Number of tables mysql will keep open

Allows to avoid reopens (expensive for some storage engines)

Needs file handlers for many storage engines

Does not affect Innodb Internal table cache.

Set at least 1024

- **thread_cache**

Number of threads MySQL can cache and reuse

Set so **threads_created** is low

32-64 is typical good value.

Innodb Settings

- **innodb_log_file_size**

Very important for write performance, reduce dirty buffer flushes

Tradeoff between performance and recovery speed

128-512M are generally good value.

- **innodb_thread_concurrency**

Number of threads which can be in Innodb kernel at once

Higher values – better resource usage but beware contention

$2 * (\text{NumCPUs} + \text{Num_disks})$ is theoretically good value

Practical values may be much lower.

- **innodb_log_buffer_size**

1-4MB is enough unless you're using large blobs.

More InnoDB Settings

- **innodb_flush_log_at_trx_commit**

Default behavior is to flush log to the disk on each transaction commit

This includes update statements outside of transactions

Often unwanted, especially for MyISAM->InnoDB migrations

Use value 2 (flush to OS cache) or 0 (do not flush at all)

InnoDB flushes logs once per second anyway

- **innodb_flush_method**

O_DIRECT to avoid double buffering

- **innodb_locks_unsafe_for_binlog**

Reduce locks if not using binary logs

Some special tuning settings

- **myisam_sort_buffer_size**

Used for rebuilding MyISAM indexes – ALTER, REPAIR, OPTIMIZE, LOAD DATA.

- **max_length_for_sort_data**

Store row data instead of row pointer in the sort file

Can help performance or can kill performance

- **myisam_stats_method**

Helps to fine tune MySQL optimizer plan selection

nulls_unequal – count all nulls as different values (default)

nulls_equal - count all nulls as same value

Time for questions

- Ask your questions now or catch me later on the conference
- pz@mysqlperformanceblog.com
- <http://www.mysqlperformanceblog.com>
- MySQL Consulting
consulting@mysqlperformanceblog.com

Presented by



O'REILLY