使用show status命令  
含义如下:  
aborted\_clients  客户端非法中断连接次数  
aborted\_connects  连接mysql失败次数  
com\_xxx xxx 命令执行次数,有很多条  
connections  连接mysql的数量  
Created\_tmp\_disk\_tables  在磁盘上创建的临时表  
Created\_tmp\_tables  在内存里创建的临时表  
Created\_tmp\_files  临时文件数  
Key\_read\_requests The number of requests to read a key block from the cache  
Key\_reads The number of physical reads of a key block from disk  
Max\_used\_connections  同时使用的连接数  
Open\_tables  开放的表  
Open\_files  开放的文件  
Opened\_tables  打开的表  
Questions  提交到server的查询数  
Sort\_merge\_passes  如果这个值很大,应该增加my.cnf中的sort\_buffer值  
Uptime  服务器已经工作的秒数

提升性能的建议:  
1.如果opened\_tables太大,应该把my.cnf中的table\_cache变大  
2.如果Key\_reads太大,则应该把my.cnf中key\_buffer\_size变大.可以用Key\_reads/Key\_read\_requests计算出cache失败率  
3.如果Handler\_read\_rnd太大,则你写的SQL语句里很多查询都是要扫描整个表,而没有发挥索引的键的作用  
4.如果Threads\_created太大,就要增加my.cnf中thread\_cache\_size的值.可以用Threads\_created/Connections计算cache命中率  
5.如果Created\_tmp\_disk\_tables太大,就要增加my.cnf中tmp\_table\_size的值,用基于内存的临时表代替基于磁盘的

## 示例

1, 查看MySQL服务器配置信息 

**代码**

1. mysql> show variables;

2, 查看MySQL服务器运行的各种状态值

**代码**

1. mysql> show global status;

3, 慢查询

**代码**

1. mysql> show variables like '%slow%';
2. +------------------+-------+
3. | Variable\_name    | Value |
4. +------------------+-------+
5. | log\_slow\_queries | OFF   |
6. | slow\_launch\_time | 2     |
7. +------------------+-------+
8. mysql> show global status like '%slow%';
9. +---------------------+-------+
10. | Variable\_name       | Value |
11. +---------------------+-------+
12. | Slow\_launch\_threads | 0     |
13. | Slow\_queries        | 279   |
14. +---------------------+-------+

配置中关闭了记录慢查询（最好是打开，方便优化），超过2秒即为慢查询，一共有279条慢查询   
  
4, 连接数 

**代码**

1. mysql> show variables like 'max\_connections';
2. +-----------------+-------+
3. | Variable\_name   | Value |
4. +-----------------+-------+
5. | max\_connections | 500   |
6. +-----------------+-------+
8. mysql> show global status like 'max\_used\_connections';
9. +----------------------+-------+
10. | Variable\_name        | Value |
11. +----------------------+-------+
12. | Max\_used\_connections | 498   |
13. +----------------------+-------+

设置的最大连接数是500，而响应的连接数是498   
  
max\_used\_connections / max\_connections \* 100% = 99.6% （理想值 ≈ 85%）   
  
5, key\_buffer\_size   
key\_buffer\_size是对MyISAM表性能影响最大的一个参数, 不过数据库中多为Innodb 

**代码**

1. mysql> show variables like 'key\_buffer\_size';
2. +-----------------+----------+
3. | Variable\_name   | Value    |
4. +-----------------+----------+
5. | key\_buffer\_size | 67108864 |
6. +-----------------+----------+
8. mysql> show global status like 'key\_read%';
9. +-------------------+----------+
10. | Variable\_name     | Value    |
11. +-------------------+----------+
12. | Key\_read\_requests | 25629497 |
13. | Key\_reads         | 66071    |
14. +-------------------+----------+

一共有25629497个索引读取请求，有66071个请求在内存中没有找到直接从硬盘读取索引，计算索引未命中缓存的概率：   
key\_cache\_miss\_rate ＝ Key\_reads / Key\_read\_requests \* 100% =0.27%   
需要适当加大key\_buffer\_size 

**代码**

1. mysql> show global status like 'key\_blocks\_u%';
2. +-------------------+-------+
3. | Variable\_name     | Value |
4. +-------------------+-------+
5. | Key\_blocks\_unused | 10285 |
6. | Key\_blocks\_used   | 47705 |
7. +-------------------+-------+

Key\_blocks\_unused表示未使用的缓存簇(blocks)数，Key\_blocks\_used表示曾经用到的最大的blocks数   
Key\_blocks\_used / (Key\_blocks\_unused + Key\_blocks\_used) \* 100% ≈ 18% （理想值 ≈ 80%）   
  
6， 临时表 

**代码**

1. mysql> show global status like 'created\_tmp%';
2. +-------------------------+---------+
3. | Variable\_name           | Value   |
4. +-------------------------+---------+
5. | Created\_tmp\_disk\_tables | 4184337 |
6. | Created\_tmp\_files       | 4124    |
7. | Created\_tmp\_tables      | 4215028 |
8. +-------------------------+---------+

每次创建临时表，Created\_tmp\_tables增加，如果是在磁盘上创建临时表，Created\_tmp\_disk\_tables也增加,Created\_tmp\_files表示MySQL服务创建的临时文件文件数：   
Created\_tmp\_disk\_tables / Created\_tmp\_tables \* 100% ＝ 99% （理想值<= 25%） 

**代码**

1. mysql> show variables where Variable\_name in ('tmp\_table\_size', 'max\_heap\_table\_size');
2. +---------------------+-----------+
3. | Variable\_name       | Value     |
4. +---------------------+-----------+
5. | max\_heap\_table\_size | 134217728 |
6. | tmp\_table\_size      | 134217728 |
7. +---------------------+-----------+

需要增加tmp\_table\_size   
  
7,open table 的情况

**代码**

1. mysql> show global status like 'open%tables%';
2. +---------------+-------+
3. | Variable\_name | Value |
4. +---------------+-------+
5. | Open\_tables   | 1024  |
6. | Opened\_tables | 1465  |
7. +---------------+-------+

Open\_tables 表示打开表的数量，Opened\_tables表示打开过的表数量，如果Opened\_tables数量过大，说明配置中 table\_cache(5.1.3之后这个值叫做table\_open\_cache)值可能太小，我们查询一下服务器table\_cache值

**代码**

1. mysql> show variables like 'table\_cache';
2. +---------------+-------+
3. | Variable\_name | Value |
4. +---------------+-------+
5. | table\_cache   | 1024  |
6. +---------------+-------+

Open\_tables / Opened\_tables \* 100% =69% 理想值 （>= 85%）   
Open\_tables / table\_cache \* 100% = 100% 理想值 (<= 95%)   
  
8, 进程使用情况

**代码**

1. mysql> show global status like 'Thread%';
2. +-------------------+-------+
3. | Variable\_name     | Value |
4. +-------------------+-------+
5. | Threads\_cached    | 31    |
6. | Threads\_connected | 239   |
7. | Threads\_created   | 2914  |
8. | Threads\_running   | 4     |
9. +-------------------+-------+

如果我们在MySQL服务器配置文件中设置了thread\_cache\_size，当客户端断开之后，服务器处理此客户的线程将会缓存起来以响应下一个客户而不是销毁（前提是缓存数未达上限）。Threads\_created表示创建过的线程数，如果发现Threads\_created值过大的话，表明 MySQL服务器一直在创建线程，这也是比较耗资源，可以适当增加配置文件中thread\_cache\_size值，查询服务器 thread\_cache\_size配置：

**代码**

1. mysql> show variables like 'thread\_cache\_size';
2. +-------------------+-------+
3. | Variable\_name     | Value |
4. +-------------------+-------+
5. | thread\_cache\_size | 32    |
6. +-------------------+-------+

9, 查询缓存(query cache)

**代码**

1. mysql> show global status like 'qcache%';
2. +-------------------------+----------+
3. | Variable\_name           | Value    |
4. +-------------------------+----------+
5. | Qcache\_free\_blocks      | 2226     |
6. | Qcache\_free\_memory      | 10794944 |
7. | Qcache\_hits             | 5385458  |
8. | Qcache\_inserts          | 1806301  |
9. | Qcache\_lowmem\_prunes    | 433101   |
10. | Qcache\_not\_cached       | 4429464  |
11. | Qcache\_queries\_in\_cache | 7168     |
12. | Qcache\_total\_blocks     | 16820    |
13. +-------------------------+----------+

Qcache\_free\_blocks：缓存中相邻内存块的个数。数目大说明可能有碎片。FLUSH QUERY CACHE会对缓存中的碎片进行整理，从而得到一个空闲块。   
Qcache\_free\_memory：缓存中的空闲内存。   
Qcache\_hits：每次查询在缓存中命中时就增大   
Qcache\_inserts：每次插入一个查询时就增大。命中次数除以插入次数就是不中比率。   
Qcache\_lowmem\_prunes：缓存出现内存不足并且必须要进行清理以便为更多查询提供空间的次数。这个数字最好长时间来看；如果这个数字在不断增长，就表示可能碎片非常严重，或者内存很少。（上面的          free\_blocks和free\_memory可以告诉您属于哪种情况）   
Qcache\_not\_cached：不适合进行缓存的查询的数量，通常是由于这些查询不是 SELECT 语句或者用了now()之类的函数。   
Qcache\_queries\_in\_cache：当前缓存的查询（和响应）的数量。   
Qcache\_total\_blocks：缓存中块的数量。   
  
我们再查询一下服务器关于query\_cache的配置：

**代码**

1. mysql> show variables like 'query\_cache%';
2. +------------------------------+----------+
3. | Variable\_name                | Value    |
4. +------------------------------+----------+
5. | query\_cache\_limit            | 33554432 |
6. | query\_cache\_min\_res\_unit     | 4096     |
7. | query\_cache\_size             | 33554432 |
8. | query\_cache\_type             | ON       |
9. | query\_cache\_wlock\_invalidate | OFF      |
10. +------------------------------+----------+

各字段的解释：   
  
query\_cache\_limit：超过此大小的查询将不缓存   
query\_cache\_min\_res\_unit：缓存块的最小大小   
query\_cache\_size：查询缓存大小   
query\_cache\_type：缓存类型，决定缓存什么样的查询，示例中表示不缓存 select sql\_no\_cache 查询   
query\_cache\_wlock\_invalidate：当有其他客户端正在对MyISAM表进行写操作时，如果查询在query cache中，是否返回cache结果还是等写操作完成再读表获取结果。   
  
query\_cache\_min\_res\_unit的配置是一柄”双刃剑”，默认是4KB，设置值大对大数据查询有好处，但如果你的查询都是小数据查询，就容易造成内存碎片和浪费。   
  
查询缓存碎片率 = Qcache\_free\_blocks / Qcache\_total\_blocks \* 100%   
  
如果查询缓存碎片率超过20%，可以用FLUSH QUERY CACHE整理缓存碎片，或者试试减小query\_cache\_min\_res\_unit，如果你的查询都是小数据量的话。   
  
查询缓存利用率 = (query\_cache\_size – Qcache\_free\_memory) / query\_cache\_size \* 100%   
  
查询缓存利用率在25%以下的话说明query\_cache\_size设置的过大，可适当减小；查询缓存利用率在80％以上而且Qcache\_lowmem\_prunes > 50的话说明query\_cache\_size可能有点小，要不就是碎片太多。   
  
查询缓存命中率 = (Qcache\_hits – Qcache\_inserts) / Qcache\_hits \* 100%   
  
示例服务器 查询缓存碎片率 ＝ 20.46％，查询缓存利用率 ＝ 62.26％，查询缓存命中率 ＝ 1.94％，命中率很差，可能写操作比较频繁吧，而且可能有些碎片。   
  
10,排序使用情况 

**代码**

1. mysql> show global status like 'sort%';
2. +-------------------+----------+
3. | Variable\_name     | Value    |
4. +-------------------+----------+
5. | Sort\_merge\_passes | 2136     |
6. | Sort\_range        | 81888    |
7. | Sort\_rows         | 35918141 |
8. | Sort\_scan         | 55269    |
9. +-------------------+----------+

Sort\_merge\_passes 包括两步。MySQL 首先会尝试在内存中做排序，使用的内存大小由系统变量 Sort\_buffer\_size 决定，如果它的大小不够把所有的记录都读到内存中，MySQL 就会把每次在内存中排序的结果存到临时文件中，等 MySQL 找到所有记录之后，再把临时文件中的记录做一次排序。这再次排序就会增加 Sort\_merge\_passes。实际上，MySQL 会用另一个临时文件来存再次排序的结果，所以通常会看到 Sort\_merge\_passes 增加的数值是建临时文件数的两倍。因为用到了临时文件，所以速度可能会比较慢，增加 Sort\_buffer\_size 会减少 Sort\_merge\_passes 和 创建临时文件的次数。但盲目的增加 Sort\_buffer\_size 并不一定能提高速度，见 How fast can you sort data with MySQL?（引自http://qroom.blogspot.com/2007/09/mysql-select-sort.html）   
  
另外，增加read\_rnd\_buffer\_size(3.2.3是record\_rnd\_buffer\_size)的值对排序的操作也有一点的好处，参见：http://www.mysqlperformanceblog.com/2007/07/24/what-exactly-is- read\_rnd\_buffer\_size/   
  
11.文件打开数(open\_files) 

**代码**

1. mysql> show global status like 'open\_files';
2. +---------------+-------+
3. | Variable\_name | Value |
4. +---------------+-------+
5. | Open\_files    | 821   |
6. +---------------+-------+
8. mysql> show variables like 'open\_files\_limit';
9. +------------------+-------+
10. | Variable\_name    | Value |
11. +------------------+-------+
12. | open\_files\_limit | 65535 |
13. +------------------+-------+

比较合适的设置：Open\_files / open\_files\_limit \* 100% <= 75％   
  
正常   
  
12。 表锁情况

**代码**

1. mysql> show global status like 'table\_locks%';
2. +-----------------------+---------+
3. | Variable\_name         | Value   |
4. +-----------------------+---------+
5. | Table\_locks\_immediate | 4257944 |
6. | Table\_locks\_waited    | 25182   |
7. +-----------------------+---------+

Table\_locks\_immediate 表示立即释放表锁数，Table\_locks\_waited表示需要等待的表锁数，如果 Table\_locks\_immediate / Table\_locks\_waited > 5000，最好采用InnoDB引擎，因为InnoDB是行锁而MyISAM是表锁，对于高并发写入的应用InnoDB效果会好些.   
  
13. 表扫描情况

**代码**

1. mysql> show global status like 'handler\_read%';
2. +-----------------------+-----------+
3. | Variable\_name         | Value     |
4. +-----------------------+-----------+
5. | Handler\_read\_first    | 108763    |
6. | Handler\_read\_key      | 92813521  |
7. | Handler\_read\_next     | 486650793 |
8. | Handler\_read\_prev     | 688726    |
9. | Handler\_read\_rnd      | 9321362   |
10. | Handler\_read\_rnd\_next | 153086384 |
11. +-----------------------+-----------+

各字段解释参见http://hi.baidu.com/thinkinginlamp/blog/item/31690cd7c4bc5cdaa144df9c.html，调出服务器完成的查询请求次数：

**代码**

1. mysql> show global status like 'com\_select';
2. +---------------+---------+
3. | Variable\_name | Value   |
4. +---------------+---------+
5. | Com\_select    | 2693147 |
6. +---------------+---------+

计算表扫描率：   
  
表扫描率 ＝ Handler\_read\_rnd\_next / Com\_select   
  
如果表扫描率超过4000，说明进行了太多表扫描，很有可能索引没有建好，增加read\_buffer\_size值会有一些好处，但最好不要超过8MB。