# MySQL 参数说明文档

## 1、参考参数

[mysqld]

innodb\_flush\_method = O\_DIRECT

user = mysql

port = 3306

socket = /data/mysql/mysql.sock

datadir = /data/mysql

open\_files\_limit = 10240

back\_log = 600

max\_connections = 1000

max\_connect\_errors = 6000

max\_allowed\_packet = 32M

sort\_buffer\_size = 2M

join\_buffer\_size = 2M

thread\_cache\_size = 300

query\_cache\_size = 0

default\_storage\_engine = InnoDB

thread\_stack = 192K

tmp\_table\_size = 256M

log\_bin = /data/mysql/mysql-bin

binlog\_cache\_size = 4M

max\_binlog\_cache\_size = 8M

max\_binlog\_size = 1024M

expire\_logs\_days = 7

key\_buffer\_size = 512M

read\_buffer\_size = 1M

read\_rnd\_buffer\_size = 16M

bulk\_insert\_buffer\_size = 64M

myisam\_sort\_buffer\_size = 128M

skip\_name\_resolve = 1

lower\_case\_table\_names = 1

server\_id = 1

innodb\_additional\_mem\_pool\_size = 16M

innodb\_buffer\_pool\_size = 2048M

innodb\_data\_file\_path = ibdata1:1024M:autoextend

innodb\_file\_io\_threads = 4

innodb\_flush\_log\_at\_trx\_commit = 2

innodb\_log\_buffer\_size = 16M

innodb\_log\_file\_size = 256M

innodb\_log\_files\_in\_group = 3

innodb\_max\_dirty\_pages\_pct = 90

innodb\_lock\_wait\_timeout = 120

character\_set\_server=utf8

innodb\_file\_per\_table=1

binlog\_format = ROW

innodb\_file\_format = Barracuda

innodb\_file\_format\_max = Barracuda

[mysql]

prompt = [\\u@\\h][\\d]>\\\_

## 2、参数说明

[mysqld]

innodb\_flush\_method = O\_DIRECT

#O\_DIRECT的设置参数是告诉系统直接将数据写入磁盘，跳过文件系统的缓存。等同于使用裸设备的效果

user = mysql

port = 3306

socket = /data/mysql/mysql.sock

datadir = /data/mysql

open\_files\_limit = 10240

#操作系统允许mysqld打开的文件的数量。

back\_log = 600

# 指在MySQL暂时停止回答新请求之前的短时间内多少个请求可以被存在堆栈中。也就是说，如果MySql的连接数达到max\_connections时，新来的请求将会被存在堆栈中，以等待某一连接释放资源，该堆栈的数量即back\_log，。默认值50。

max\_connections = 1000

#MySQL允许最大的进程连接数，如果经常出现Too Many Connections的错误提示，则需要增大此值。

max\_connect\_errors = 6000

#每个主机的连接请求异常中断的最大次数，当超过该次数，MYSQL服务器将禁止host的连接请求，直到mysql服务器重启或通过flush hosts命令清空此host的相关信息。

max\_allowed\_packet = 32M

#在网络传输中一次消息传输量的最大值。必须设置1024的倍数。

sort\_buffer\_size = 2M

#每个排序线程分配的缓冲区的大小。增加该值可以加快ORDER BY或GROUP BY操作。

join\_buffer\_size = 2M

#用于完全联接的缓冲区的大小(当不使用索引的时候使用联接操作)。一般情况获得快速联接的最好方法是添加索引。当增加索引时不可能通过增加join\_buffer\_size值来获得快速完全联接。将为两个表之间的每个完全联接分配联接缓冲区。对于多个表之间不使用索引的复杂联接，需要多联接缓冲区。

thread\_cache\_size = 300

#服务器线程缓存这个值表示可以重新利用保存在缓存中线程的数量,当断开连接时如果缓存中还有空间,那么客户端的线程将被放到缓存中,如果线程重新被请求，那么请求将从缓存中读取,如果缓存中是空的或者是新的请求，那么这个线程将被重新创建,如果有很多新的线程，增加这个值可以改善系统性能.通过比较Connections和Threads\_created状态的变量，可以看到这个变量的作用。设置规则如下：2GB配置为16，3GB配置为32，4GB或更高内存，可配置更大。

query\_cache\_size = 0

#为缓存查询结果分配的内存的数量。默认值是0，即禁用查询缓存。

default\_storage\_engine = InnoDB

thread\_stack = 192K

#每个线程的堆栈大小。用crash-me测试检测出的许多限制取决于该值。 默认值足够大，可以满足普通操作。

tmp\_table\_size = 256M

#tmp\_table\_size的默认大小是32M。如果你做很多高级GROUP BY查询，增加tmp\_table\_size值。如果超过该值，则会将临时表写入磁盘。

Log\_bin = /data/mysql/mysql-bin

#二进制日志文件。将更改数据的所有查询记入该文件。用于备份和复制。建议指定一个文件名,否则MySQL使用 host\_name -bin作为日志文件基本名

binlog\_cache\_size = 4M

#在事务过程中容纳二进制日志SQL语句的缓存大小。二进制日志缓存是服务器支持事务存储引擎并且服务器启用了二进制日志(--log-bin选项)的前提下为每个客户端分配的内存。如果你经常使用大的，多语句事务，你可以增加该值以获得更有的性能。Binlog\_cache\_use和Binlog\_cache\_disk\_use状态变量可以用来调整该变量的大小.

max\_binlog\_cache\_size = 8M

#如果多语句事务需要更大的内存，你会得到错误Multi-statement transaction required more than 'max\_binlog\_cache\_size' bytes of storage。

max\_binlog\_size = 1024M

#如果二进制日志写入的内容超出给定值，日志就会发生滚动。你不能将该变量设置为大于1GB或小于4096字节。

expire\_logs\_days = 7

#二进制日志自动删除的天数。默认值为0,表示“没有自动删除”

key\_buffer\_size = 512M

# key\_buffer\_size 指定用于索引的缓冲区大小，增加它可得到更好处理的索引(对所有读和多重写)，到你能负担得起那样多。如果你使它太大，系统将开始换页并且真的变慢了。

对于内存在4GB左右的服务器该参数可设置为384M或512M。

read\_buffer\_size = 1M

#每个线程连续扫描时为扫描的每个表分配的缓冲区的大小(字节)。如果进行多次连续扫描，可能需要增加该值， 默认值为131072。

read\_rnd\_buffer\_size = 16M

#MySql的随机读（查询操作）缓冲区大小。当按任意顺序读取行时(例如，按照排序顺序)，将分配一个随机读缓存区。进行排序查询时，MySql会首先扫描一遍该缓冲，以避免磁盘搜索，提高查询速度，如果需要排序大量数据，可适当调高该值。但MySql会为每个客户连接发放该缓冲空间，所以应尽量适当设置该值，以避免内存开销过大。

bulk\_insert\_buffer\_size = 64M

#MyISAM 使用专用树状缓存来使INSERT ... SELECT、INSERT ... VALUES (...)、(...)、 ...和LOAD DATA INFILE的大块插入更快。该变量用每线程的字节数限制缓存树的大小。将它设置为0禁用优化。注释：只有向非空表添加数据时才使用该缓存。 默认值是8MB。

myisam\_sort\_buffer\_size = 128M

#MyISAM表发生变化时重新排序所需的缓冲

skip\_name\_resolve = 1

#不要解析正检查客户端连接的主机名。只使用IP号。如果你使用该项， 授权表中的所有Host列值必须为IP号或localhost。

lower\_case\_table\_names = 1

#如果设置为1,表名用小写保存到硬盘上，并且表名比较时不对大小写敏感。如果设置为2，按照指定的保存表名，但按照小写来比较。该选项还适合数据库名和表的别名。

server\_id = 1

#用于主复制服务器和从复制服务器。

innodb\_additional\_mem\_pool\_size = 16M

#设置InnoDB存储的数据目录信息和其它内部数据结构的内存池大小，类似于Oracle的library cache。

innodb\_buffer\_pool\_size = 2048M

#InnoDB用来缓存它的数据和索引的内存缓冲区的大小。你把这个值设得越高，访问表中数据需要得磁盘I/O越少。在专用服务器中建议设置物理内存的70%-80%。尽管如此，还是不要把它设置得太大，因为对物理内存的竞争可能在操作系统上导致内存调度。

innodb\_data\_file\_path = ibdata1:1024M:autoextend

#到单独数据文件和它们尺寸的路径。通过把innodb\_data\_home\_dir连接到这里指定的每个路径，到每个数据文件的完整目录路径可被获得。文件大小通过给尺寸值尾加M或G以MB或者GB（1024MB）为单位被指定。文件尺寸的和至少是10MB。

innodb\_file\_io\_threads = 4

#InnoDB中文件I/O线程的数量。正常地，这个参数是用 默认的，默认值是4，但是大数值对Windows磁盘I/O有益。在Unix上，增加这个数没有效果，InnoDB总是使用默认值。

innodb\_flush\_log\_at\_trx\_commit = 2

# 如果将此参数设置为1，将在每次提交事务后将日志写入磁盘。为提供性能，可以设置为0或2，但要承担在发生故障时丢失数据的风险。设置为0表示事务日志写入日志文件，而日志文件每秒刷新到磁盘一次。设置为2表示事务日志将在提交时写入日志，但日志文件每次刷新到磁盘一次。

innodb\_log\_buffer\_size = 16M

#InnoDB用来往磁盘上的日志文件写操作的缓冲区的大小。 默认的是1MB。一个大的日志缓冲允许大型事务运行而不需要在事务提交之前往磁盘写日志。因此，如果你有大型事务，使日志缓冲区更大以节约磁盘I/O。

innodb\_log\_file\_size = 256M

#此参数确定数据日志文件的大小，以M为单位，更大的设置可以提高性能，但也会增加恢复故障数据库所需的时间

innodb\_log\_files\_in\_group = 3

#为提高性能，MySQL可以以循环方式将日志文件写到多个文件。推荐设置为3M

innodb\_max\_dirty\_pages\_pct = 90

#这是一个范围从0到100的整数。默认是90。InnoDB中的主线程试着从缓冲池写页面，使得脏页（没有被写的页面）的百分比不超过这个值。

innodb\_lock\_wait\_timeout = 120

#InnoDB事务在被回滚之前可以等待一个锁定的超时秒数。InnoDB在它自己的 锁定表中自动检测事务死锁并且回滚事务。InnoDB用LOCK TABLES语句注意到锁定设置。默认值是50秒。

character\_set\_server=utf8

innodb\_file\_per\_table=1

#启用独立表空间

binlog\_format = ROW

#日志格式

innodb\_file\_format = Barracuda

#Antelope是innodb-base的文件格式，Barracude是innodb-plugin后引入的文件格式，同时Barracude也支持Antelope文件格式

innodb\_file\_format\_max = Barracuda

[mysql]

prompt = [\\u@\\h][\\d]>\\\_