


```
port = @MYSQL_TCP_PORT@
socket = @MYSQL_UNIX_ADDR@
# back_log 是操作系统在监听队列中所能保持的连接数,
# 队列保存了在 MySQL 连接管理器线程处理之前的连接.
# 如果你有非常高的连接率并且出现"connection refused" 报错,
# 你就应该增加此处的值.
# 检查你的操作系统文档来获取这个变量的最大值.
# 如果将 back_log 设定到比你操作系统限制更高的值, 将会没有效果
back_log = 50
# 不在 TCP/IP 端口上进行监听.
# 如果所有的进程都是在同一台服务器连接到本地的 mysqld,
# 这样设置将是增强安全的方法
# 所有 mysqld 的连接都是通过 Unix sockets 或者命名管道进行的.
# 注意在 windows 下如果没有打开命名管道选项而只是用此项
# (通过 "enable-named-pipe" 选项) 将会导致 mysql 服务没有任何作用!
#skip-networking
# MySQL 服务所允许的同时会话数的上限
# 其中一个连接将被 SUPER 权限保留作为管理员登录.
# 即便已经达到了连接数的上限.
max_connections = 100
# 每个客户端连接最大的错误允许数量, 如果达到了此限制.
# 这个客户端将会被 MySQL 服务阻止直到执行了"FLUSH HOSTS" 或者服务重启
# 非法的密码以及其他在链接时的错误会增加此值.
# 查看 "Aborted_connects" 状态来获取全局计数器.
max_connect_errors = 10
# 所有线程所打开表的数量.
# 增加此值就增加了 mysqld 所需要的文件描述符的数量
# 这样你需要确认在 [mysqld_safe] 中 "open-files-limit" 变量设置打开文件数量允许至少 4096
table_cache = 2048
# 允许外部文件级别的锁. 打开文件锁会对性能造成负面影响
# 所以只有在你在同样的文件上运行多个数据库实例时才使用此选项 (注意仍会有其他约束!)
# 或者你在文件层面上使用了其他一些软件依赖来锁定 MyISAM 表
#external-locking
# 服务所能处理的请求包的最大大小以及服务所能处理的最大的请求大小 (当与大的 BLOB 字段一起工作时相当必要)
# 每个连接独立的大小. 大小动态增加
max_allowed_packet = 16M
# 在一个事务中 binlog 为了记录 SQL 状态所持有的 cache 大小
# 如果你经常使用大的, 多声明的事务, 你可以增加此值来获取更大的性能.
# 所有从事务来的状态都将被缓冲在 binlog 缓冲中然后在提交后一次性写入到 binlog 中
# 如果事务比此值大, 会使用磁盘上的临时文件来替代.
# 此缓冲在每个连接的事务第一次更新状态时被创建
```

```
binlog_cache_size = 1M
# 独立的内存表所允许的最大容量.
# 此选项为了防止意外创建一个超大的内存表导致耗尽所有的内存资源.
max_heap_table_size = 64M
# 排序缓冲被用来处理类似 ORDER BY 以及 GROUP BY 队列所引起的排序
# 如果排序后的数据无法放入排序缓冲,
# 一个用来替代的基于磁盘的合并分类会被使用
# 查看 "Sort_merge_passes" 状态变量.
# 在排序发生时由每个线程分配
sort_buffer_size = 8M
# 此缓冲被使用来优化全联合 (full JOINs 不带索引的联合).
# 类似的联合在绝大多数情况下有非常糟糕的性能表现,
# 但是将此值设大能够减轻性能影响.
# 通过 "Select_full_join" 状态变量查看全联合的数量
# 当全联合发生时, 在每个线程中分配
join_buffer_size = 8M
# 我们在 cache 中保留多少线程用于重用
# 当一个客户端断开连接后, 如果 cache 中的线程还少于 thread_cache_size,
# 则客户端线程被放入 cache 中.
# 这可以在你需要大量新连接的时候极大的减少线程创建的开销
# (一般来说如果你有好的线程模型的话, 这不会有明显的性能提升.)
thread_cache_size = 8
# 此允许应用程序给予线程系统一个提示在同一时间给予渴望被运行的线程的数量.
# 此值只对于支持 thread_concurrency() 函数的系统有意义(例如 Sun Solaris).
# 你可以尝试使用 [CPU 数量]*(2..4) 来作为 thread_concurrency 的值
thread_concurrency = 8
# 查询缓冲常被用来缓冲 SELECT 的结果并且在下一次同样查询的时候不再执行直接返回结果.
# 打开查询缓冲可以极大的提高服务器速度, 如果你有大量的相同的查询并且很少修改表.
# 查看 "Qcache_lowmem_prunes" 状态变量来检查是否当前值对于你的负载来说是否足够高.
# 注意: 在你表经常变化的情况下或者如果你的查询原文每次都不同,
# 查询缓冲也许引起性能下降而不是性能提升.
query_cache_size = 64M
# 只有小于此设定值的结果才会被缓冲
# 此设置用来保护查询缓冲, 防止一个极大的结果集将其他所有的查询结果都覆盖.
query_cache_limit = 2M
# 被全文检索索引的最小的字长.
# 你也许希望减少它, 如果你需要搜索更短字的时候.
# 注意在你修改此值之后,
# 你需要重建你的 FULLTEXT 索引
ft_min_word_len = 4
# 如果你的系统支持 memlock() 函数, 你也许希望打开此选项用以让运行中的 mysql 在在
```

[illegible]

```
# 一般此项最好打开。
# 打开此项会记录使得那些没有使用索引的查询也被作为到慢速查询附加到慢速日志里
log_long_format
# 此目录被 MySQL 用来保存临时文件。例如，
# 它被用来处理基于磁盘的大型排序，和内部排序一样。
# 以及简单的临时表。
# 如果你不创建非常大的临时文件，将其放置到 swapfs/tmpfs 文件系统上也许比较好
# 另一种选择是你也可以将其放置在独立的磁盘上。
# 你可以使用“;”来放置多个路径
# 他们会按照 round-robin 方法被轮询使用。
tmpdir = /tmp
# ***复制有关的设置
# 唯一的服务器识别号，数值位于 1 到 2^32-1 之间。
# 此值在 master 和 slave 上都需要设置。
# 如果 “master-host” 没有被设置，则默认为 1，但是如果忽略此选项，MySQL
不会作为 master 生效。
server-id = 1
# 复制的 Slave（去掉 master 段的注释来使其生效）
#
# 为了配置此主机作为复制的 slave 服务器，你可以选择两种方法：
#
# 1) 使用 CHANGE MASTER TO 命令（在我们的手册中有完整描述） -
#&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; 语法如下：
#
#&nbsp; &nbsp; &nbsp; &nbsp; CHANGE MASTER TO MASTER_HOST=, MASTER_PORT=,
#&nbsp; &nbsp; &nbsp; &nbsp; MASTER_USER=, MASTER_PASSWORD= ;
#
#&nbsp; &nbsp; &nbsp; &nbsp; 你需要替换掉 , ,&nbsp;&nbsp;&nbsp;等被尖括号包围的字段以及使用 master
的端口号替换（默认 3306）。
#
#&nbsp; &nbsp; &nbsp; &nbsp; 例子：
#
#&nbsp; &nbsp; &nbsp; &nbsp; CHANGE MASTER TO MASTER_HOST='125.564.12.1', MASTER_PORT=3306,
#&nbsp; &nbsp; &nbsp; &nbsp; MASTER_USER='joe', MASTER_PASSWORD='secret';
#
# 或者
#
# 2) 设置以下的变量。不论如何，在你选择这种方法的情况下，然后第一次启动复制（甚至不成功的情况下），
#&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; 例如如果你输入错密码在 master-password 字段并且 slave
无法连接），
#&nbsp; &nbsp; &nbsp; &nbsp; slave 会创建一个 master.info 文件，并且之后任何对于包含在此文件内
的参数变化都会被忽略
#&nbsp; &nbsp; &nbsp; &nbsp; 并且由 master.info 文件内的内容覆盖。除非你关闭 slave 服务，删除
```

```

master.info 并且重启 slave 服务.
# 由于这个原因, 你也许不想碰一下的配置 (注释掉的) 并且使用 CHANGE
MASTER TO (查看上面) 来代替
#
# 所需要的唯一 id 号位于 2 和 2^32 - 1 之间
# (并且和 master 不同)
# 如果 master-host 被设置了. 则默认值是 2
# 但是如果省略, 则不会生效
#server-id = 2
#
# 复制结构中的 master - 必须
#master-host =
#
# 当连接到 master 上时 slave 所用来认证的用户名 - 必须
#master-user =
#
# 当连接到 master 上时 slave 所用来认证的密码 - 必须
#master-password =
#
# master 监听的端口.
# 可选 - 默认是 3306
#master-port =
# 使得 slave 只读. 只有用户拥有 SUPER 权限和在上面的 slave 线程能够修改数据.
# 你可以使用此项去保证没有应用程序会意外的修改 slave 而不是 master 上的数据
#read_only
#*** MyISAM 相关选项
# 关键词缓冲的大小, 一般用来缓冲 MyISAM 表的索引块.
# 不要将其设置大于你可用内存的 30%,
# 因为一部分内存同样被 OS 用来缓冲行数据
# 甚至在你并不使用 MyISAM 表的情况下, 你也需要仍旧设置起 8-64M 内存由于它同样会
被内部临时磁盘表使用.
key_buffer_size = 32M
# 用来做 MyISAM 表全表扫描的缓冲大小.
# 当全表扫描需要时, 在对应线程中分配.
read_buffer_size = 2M
# 当在排序之后, 从一个已经排序好的序列中读取行时, 行数据将从这个缓冲中读取来防止
磁盘寻道.
# 如果你增高此值, 可以提高很多 ORDER BY 的性能.
# 当需要时由每个线程分配
read_rnd_buffer_size = 16M
# MyISAM 使用特殊的类似树的 cache 来使得突发插入
# (这些插入是, INSERT ... SELECT, INSERT ... VALUES (...), (...), ..., 以及 LOAD
DATA
# INFILE) 更快. 此变量限制每个进程中缓冲树的字节数.

```

```
# 设置为 0 会关闭此优化.
# 为了最优化不要将此值设置大于 "key_buffer_size"。
# 当突发插入被检测到时此缓冲将被分配。
bulk_insert_buffer_size = 64M
# 此缓冲当 MySQL 需要在 REPAIR, OPTIMIZE, ALTER 以及 LOAD DATA INFILE 到一个空表
# 中引起重建索引时被分配。
# 这在每个线程中被分配。所以在设置大值时需要小心。
myisam_sort_buffer_size = 128M
# MySQL 重建索引时所允许的最大临时文件的大小（当 REPAIR, ALTER TABLE 或者 LOAD
# DATA INFILE）。
# 如果文件大小比此值更大, 索引会通过键值缓冲创建(更慢)
myisam_max_sort_file_size = 10G
# 如果被用来更快的索引创建索引所使用临时文件大于制定的值, 那就使用键值缓冲方法。
# 这主要用来强制在大表中长字符串键去使用慢速的键值缓冲方法来创建索引。
myisam_max_extra_sort_file_size = 10G
# 如果一个表拥有超过一个索引, MyISAM 可以通过并行排序使用超过一个线程去修复他
# 们。
# 这对于拥有多个 CPU 以及大量内存情况的用户, 是一个很好的选择。
myisam_repair_threads = 1
# 自动检查和修复没有适当关闭的 MyISAM 表。
myisam_recover
# 默认关闭 Federated
skip-federated
# *** BDB 相关选项 ***
# 如果你运行的 MySQL 服务有 BDB 支持但是你不准备使用的时候使用此选项。这会节省内
# 存并且可能加速一些事。
skip-bdb
# *** INNODB 相关选项 ***
# 如果你的 MySQL 服务包含 InnoDB 支持但是并不打算使用的话,
# 使用此选项会节省内存以及磁盘空间, 并且加速某些部分
#skip-innodb
# 附加的内存池被 InnoDB 用来保存 metadata 信息
# 如果 InnoDB 为此目的需要更多的内存, 它会开始从 OS 这里申请内存。
# 由于这个操作在大多数现代操作系统上已经足够快, 你一般不需要修改此值。
# SHOW INNODB STATUS 命令会显示当先使用的数量。
innodb_additional_mem_pool_size = 16M
# InnoDB 使用一个缓冲池来保存索引和原始数据, 不像 MyISAM。
# 这里你设置越大, 你在存取表里面数据时所需要的磁盘 I/O 越少。
# 在一个独立使用的数据库服务器上, 你可以设置这个变量到服务器物理内存大小的 80%
# 不要设置过大, 否则, 由于物理内存的竞争可能导致操作系统的换页颠簸。
# 注意在 32 位系统上你每个进程可能被限制在 2-3.5G 用户层面内存限制,
# 所以不要设置的太高。
innodb_buffer_pool_size = 2G
# InnoDB 将数据保存在一个或者多个数据文件中成为表空间。
```

```
# 如果你只有单个逻辑驱动保存你的数据, 一个单个的自增文件就足够好了.
# 其他情况下, 每个设备一个文件一般都是个好的选择.
# 你也可以配置 InnoDB 来使用裸盘分区 - 请参考手册来获取更多内容
innodb_data_file_path = ibdata1:10M:autoextend
# 设置此选项如果你希望 InnoDB 表空间文件被保存在其他分区.
# 默认保存在 MySQL 的 datadir 中.
#innodb_data_home_dir =
# 用来同步 IO 操作的 IO 线程的数量. This value is
# 此值在 Unix 下被硬编码为 4, 但是在 Windows 磁盘 I/O 可能在一个大数值下表现的更好.
innodb_file_io_threads = 4
# 如果你发现 InnoDB 表空间损坏, 设置此值为一个非零值可能帮助你导出你的表.
# 从 1 开始并且增加此值知道你能够成功的导出表.
#innodb_force_recovery=1
# 在 InnoDB 核心内的允许线程数量.
# 最优值依赖于应用程序, 硬件以及操作系统的调度方式.
# 过高的值可能导致线程的互斥颠簸.
innodb_thread_concurrency = 16
# 如果设置为 1, InnoDB 会在每次提交后刷新 (fsync) 事务日志到磁盘上,
# 这提供了完整的 ACID 行为.
# 如果你愿意对事务安全折衷, 并且你正在运行一个小的食物, 你可以设置此值到 0 或者 2
# 来减少由事务日志引起的磁盘 I/O
# 0 代表日志只大约每秒写入日志文件并且日志文件刷新到磁盘.
# 2 代表日志写入日志文件在每次提交后, 但是日志文件只有大约每秒才会刷新到磁盘上.
innodb_flush_log_at_trx_commit = 1
# 加速 InnoDB 的关闭. 这会阻止 InnoDB 在关闭时做全清除以及插入缓冲合并.
# 这可能极大增加关机时间, 但是取而代之的是 InnoDB 可能在下次启动时做这些操作.
#innodb_fast_shutdown
# 用来缓冲日志数据的缓冲区的大小.
# 当此值快满时, InnoDB 将必须刷新数据到磁盘上.
# 由于基本上每秒都会刷新一次, 所以没有必要将此值设置的太大 (甚至对于长事务而言)
innodb_log_buffer_size = 8M
# 在日志组中每个日志文件的大小.
# 你应该设置日志文件总合大小到你缓冲池大小的 25%~100%
# 来避免在日志文件覆写上不必要的缓冲池刷新行为.
# 不论如何, 请注意一个大的日志文件大小会增加恢复进程所需要的时间.
innodb_log_file_size = 256M
# 在日志组中的文件总数.
# 通常来说 2~3 是比较好的.
innodb_log_files_in_group = 3
# InnoDB 的日志文件所在位置. 默认是 MySQL 的 datadir.
# 你可以将其指定到一个独立的硬盘上或者一个 RAID1 卷上来提高其性能
#innodb_log_group_home_dir
# 在 InnoDB 缓冲池中最大允许的脏页面的比例.
# 如果达到限额, InnoDB 会开始刷新他们防止他们妨碍到干净数据页面.
```



```
# 这是一个软限制, 不被保证绝对执行.
innodb_max_dirty_pages_pct = 90
# InnoDB 用来刷新日志的方法.
# 表空间总是使用双重写入刷新方法
# 默认值是 "fdatasync", 另一个是 "O_DSYNC".
innodb_flush_method=O_DSYNC
# 在被回滚前, 一个 InnoDB 的事务应该等待一个锁被批准多久.
# InnoDB 在其拥有的锁表中自动检测事务死锁并且回滚事务.
# 如果你使用 LOCK TABLES 指令, 或者在同样事务中使用除了 InnoDB 以外的其他事务安全的存储引擎
# 那么一个死锁可能发生而 InnoDB 无法注意到.
# 这种情况下这个 timeout 值对于解决这种问题就非常有帮助.
innodb_lock_wait_timeout = 120
[mysqldump]
# 不要在将内存中的整个结果写入磁盘之前缓存. 在导出非常巨大的表时需要此项
quick
max_allowed_packet = 16M
[mysql]
no-auto-rehash
# 仅仅允许使用键值的 UPDATEs 和 DELETEs .
#safe-updates
[isamchk]
key_buffer = 512M
sort_buffer_size = 512M
read_buffer = 8M
write_buffer = 8M
[myisamchk]
key_buffer = 512M
sort_buffer_size = 512M
read_buffer = 8M
write_buffer = 8M
[mysqlhotcopy]
interactive-timeout
[mysqld_safe]
# 增加每个进程的可打开文件数量.
# 警告: 确认你已经将全系统限制设置的足够高!
# 打开大量表需要将此值设高
open-files-limit = 8192
```