[root@localhost redis]# vi /usr/local/redis/etc/redis.conf

#是否以后台进程运行，默认为no，如果需要以后台进程运行则改为yes

daemonize no

#如果以后台进程运行的话，就需要指定pid，你可以在此自定义redis.pid文件的位置。

pidfile /var/run/redis.pid

#接受连接的端口号，如果端口是0则redis将不会监听TCP socket连接

port 6379

# If you want you can bind a single interface, if the bind option is not

# specified all the interfaces will listen for incoming connections.

#

# bind 127.0.0.1

# Specify the path for the unix socket that will be used to listen for

# incoming connections. There is no default, so Redis will not listen

# on a unix socket when not specified.

#

# unixsocket /tmp/redis.sock

# unixsocketperm 755

#连接超时时间，单位秒。(0 to disable)？

timeout 300000000

#日志级别，默认是verbose（详细），各种日志级别：

#debug:很详细的信息，适合开发和测试

#verbose:包含许多不太有用的信息，但比debug要清爽一些（many rarely useful info, but not a mess like #the debug level）

#notice:比较适合生产环境

#warning:警告信息

loglevel verbose

#指定log文件的名字，默认是空。stdout会让redis把日志输出到标准输出。但是如果使用stdout而又以后台进#程的方式运行redis，则日志会输出到/dev/null。请改为需要的日志名

logfile ""

#'syslog-enabled'设置为yes会把日志输出到系统日志，默认是no

# syslog-enabled no

#指定syslog的标示符，如果'syslog-enabled'是no，则这个选项无效。

# syslog-ident redis

#指定syslog 设备（facility), 必须是USER或者LOCAL0到LOCAL7.

# syslog-facility local0

#设置数据库数目。默认的数据库是DB 0。可以通过SELECT <dbid>来选择一个数据库，dbid是[0,'databases'-1]的数字

databases 16

################## 快照#################################

#

# 硬盘上保存数据:

#

# save <seconds> <changes>

#

# <seconds>和<changes>都满足时就会触发数据保存动作。

#

#

# 以下面的例子来说明：

# 过了900秒并且有1个key发生了改变 就会触发save动作

# 过了300秒并且有10个key发生了改变 就会触发save动作

# 过了60秒并且至少有10000个key发生了改变 也会触发save动作

#

# 注意：如果你不想让redis自动保存数据，那就把下面的配置注释掉！

save 900 1

save 300 10

save 60 10000

#存储数据时是否压缩数据。默认是yes。

rdbcompression yes

# 保存dump数据的文件名

dbfilename dump.rdb

# 工作目录.

#

# 数据会被持久化到这个目录下的‘dbfilename’指定的文件中。

#

#

# 注意，这里指定的必须是目录而不能是文件。

dir ./

######## REPLICATION（复制，冗余）#################################

# Master-Slave replication. 使用slaveof把一个 Redis 实例设置成为另一个Redis server的从库（热备）. 注意： #配置只对当前slave有效。

# 因此可以把某个slave配置成使用不同的时间间隔来保存数据或者监听其他端口等等。

#命令格式：

# slaveof <masterip> <masterport>

#如果master有密码保护，则在slave与master进行数据同步之前需要进行密码校验，否则master会拒绝slave的请#求。

#

# masterauth <master-password>

#当slave丢失与master的连接时，或者slave仍然在于master进行数据同步时（还没有与master保持一致），#slave可以有两种方式来响应客户端请求：

#

# 1) 如果 slave-serve-stale-data 设置成 'yes' (the default) slave会仍然响应客户端请求,此时可能会有问题。

#

# 2) 如果 slave-serve-stale data设置成 'no' slave会返回"SYNC with master in progress"这样的错误信息。 但 INFO 和SLAVEOF命令除外。

#

slave-serve-stale-data yes

############### 安全 ###################################

# 需要客户端在执行任何命令之前指定 AUTH <PASSWORD>

#

# requirepass foobared

# 命令重命名.

#

#

# 例如:

#

# rename-command CONFIG b840fc02d524045429941cc15f59e41cb7be6c52

#

# 同样可以通过把一个命令重命名为空串来彻底kill掉这个命令，比如：

#

# rename-command CONFIG ""

#################### 限制 ####################################

# 设置最大连接数. 默认没有限制, '0' 意味着不限制.

#

# maxclients 128

#最大可使用内存。如果超过，Redis会试图删除EXPIRE集合中的keys，具体做法是：Redis会试图释放即将过期的#keys，而保护还有很长生命周期的keys。

#

#如果这样还不行，Redis就会报错，但像GET之类的查询请求还是会得到响应。

#

#警告：如果你想把Redis视为一个真正的DB的话，那不要设置<maxmemory>,只有你只想把Redis作为cache或者

#有状态的server（'state' server)时才需要设置。

#

# maxmemory <bytes>

#内存清理策略：如果达到了maxmemory，你可以采取如下动作：

#

# volatile-lru -> 使用LRU算法来删除过期的set

# allkeys-lru -> 删除任何遵循LRU算法的key

# volatile-random ->随机地删除过期set中的key

# allkeys->random -> 随机地删除一个key

# volatile-ttl -> 删除最近即将过期的key（the nearest expire time (minor TTL)）

# noeviction -> 根本不过期，写操作直接报错

#

#

# 默认策略:

#

# maxmemory-policy volatile-lru

# 对于处理redis内存来说，LRU和minor TTL算法不是精确的，而是近似的（估计的）算法。所以我们会检查某些样本#来达到内存检查的目的。默认的样本数是3，你可以修改它。

#

# maxmemory-samples 3

################# APPEND ONLY MODE ###############################

#默认情况下，Redis会异步的把数据保存到硬盘。如果你的应用场景允许因为系统崩溃等极端情况而导致最新数据丢失#的话，那这种做法已经很ok了。否则你应该打开‘append only’模式，开启这种模式后，Redis会在#appendonly.aof文件中添加每一个写操作，这个文件会在Redis启动时被读取来在内存中重新构建数据集。

#

#注意：如果你需要，你可以同时开启‘append only’模式和异步dumps模式（你需要注释掉上面的‘save’表达式来禁#止dumps），这种情况下，Redis重建数据集时会优先使用appendonly.aof而忽略dump.rdb

#

appendonly no

# append only 文件名 (默认: "appendonly.aof")

# appendfilename appendonly.aof

# 调用fsync()函数通知操作系统立刻向硬盘写数据

#

# Redis支持3中模式:

#

# no:不fsync, 只是通知OS可以flush数据了，具体是否flush取决于OS.性能更好.

# always: 每次写入append only 日志文件后都会fsync . 性能差，但很安全.

# everysec: 没间隔1秒进行一次fsync. 折中.

#

# 默认是 "everysec"

# appendfsync always

appendfsync everysec

# appendfsync no

# 当AOF fsync策略被设置为always或者everysec并且后台保存进程（saving process)正在执行大量I/O操作时

# Redis可能会在fsync()调用上阻塞过长时间

#

no-appendfsync-on-rewrite no

# append only 文件的自动重写

# 当AOF 日志文件即将增长到指定百分比时，Redis可以通过调用BGREWRITEAOF 来自动重写append only文件。

#

# 它是这么干的：Redis会记住最近一次重写后的AOF 文件size。然后它会把这个size与当前size进行比较，如果当前# size比指定的百分比大，就会触发重写。同样，你需要指定AOF文件被重写的最小size，这对避免虽然百分比达到了# 但是实际上文件size还是很小（这种情况没有必要重写）却导致AOF文件重写的情况很有用。

#

#

# auto-aof-rewrite-percentage 设置为 0 可以关闭AOF重写功能

auto-aof-rewrite-percentage 100

auto-aof-rewrite-min-size 64mb

################## SLOW LOG ###################################

# Redis slow log用来记录超过指定执行时间的查询。

#

# 你可以指定两个参数：一个是慢查询的阀值，单位是毫秒；另外一个是slow log的长度，相当于一个队列。

# 负数则关闭slow log，0则会导致每个命令都被记录

slowlog-log-slower-than 10000

# 不设置会消耗过多内存，所以还是要设置一下。可以使用SLOWLOG RESET命令来回收slow log使用的内存

slowlog-max-len 1024

################ 虚拟内存 ###############################

#使用redis 就别用虚拟内存了，绝对不是一个好主意，加个机器吧，所以这里不翻译啦！！

### WARNING! Virtual Memory is deprecated in Redis 2.4

### The use of Virtual Memory is strongly discouraged.

# Virtual Memory allows Redis to work with datasets bigger than the actual

# amount of RAM needed to hold the whole dataset in memory.

# In order to do so very used keys are taken in memory while the other keys

# are swapped into a swap file, similarly to what operating systems do

# with memory pages.

#

# To enable VM just set 'vm-enabled' to yes, and set the following three

# VM parameters accordingly to your needs.

vm-enabled no

# vm-enabled yes

# This is the path of the Redis swap file. As you can guess, swap files

# can't be shared by different Redis instances, so make sure to use a swap

# file for every redis process you are running. Redis will complain if the

# swap file is already in use.

#

# The best kind of storage for the Redis swap file (that's accessed at random)

# is a Solid State Disk (SSD).

#

# \*\*\* WARNING \*\*\* if you are using a shared hosting the default of putting

# the swap file under /tmp is not secure. Create a dir with access granted

# only to Redis user and configure Redis to create the swap file there.

vm-swap-file /tmp/redis.swap

# vm-max-memory configures the VM to use at max the specified amount of

# RAM. Everything that deos not fit will be swapped on disk \*if\* possible, that

# is, if there is still enough contiguous space in the swap file.

#

# With vm-max-memory 0 the system will swap everything it can. Not a good

# default, just specify the max amount of RAM you can in bytes, but it's

# better to leave some margin. For instance specify an amount of RAM

# that's more or less between 60 and 80% of your free RAM.

vm-max-memory 0

# Redis swap files is split into pages. An object can be saved using multiple

# contiguous pages, but pages can't be shared between different objects.

# So if your page is too big, small objects swapped out on disk will waste

# a lot of space. If you page is too small, there is less space in the swap

# file (assuming you configured the same number of total swap file pages).

#

# If you use a lot of small objects, use a page size of 64 or 32 bytes.

# If you use a lot of big objects, use a bigger page size.

# If unsure, use the default :)

vm-page-size 32

# Number of total memory pages in the swap file.

# Given that the page table (a bitmap of free/used pages) is taken in memory,

# every 8 pages on disk will consume 1 byte of RAM.

#

# The total swap size is vm-page-size \* vm-pages

#

# With the default of 32-bytes memory pages and 134217728 pages Redis will

# use a 4 GB swap file, that will use 16 MB of RAM for the page table.

#

# It's better to use the smallest acceptable value for your application,

# but the default is large in order to work in most conditions.

vm-pages 134217728

# Max number of VM I/O threads running at the same time.

# This threads are used to read/write data from/to swap file, since they

# also encode and decode objects from disk to memory or the reverse, a bigger

# number of threads can help with big objects even if they can't help with

# I/O itself as the physical device may not be able to couple with many

# reads/writes operations at the same time.

#

# The special value of 0 turn off threaded I/O and enables the blocking

# Virtual Memory implementation.

vm-max-threads 4

################高级配置###############################

# Hashes are encoded in a special way (much more memory efficient) when they

# have at max a given numer of elements, and the biggest element does not

# exceed a given threshold. You can configure this limits with the following

# configuration directives.

hash-max-zipmap-entries 512

hash-max-zipmap-value 64

# Similarly to hashes, small lists are also encoded in a special way in order

# to save a lot of space. The special representation is only used when

# you are under the following limits:

list-max-ziplist-entries 512

list-max-ziplist-value 64

# Sets have a special encoding in just one case: when a set is composed

# of just strings that happens to be integers in radix 10 in the range

# of 64 bit signed integers.

# The following configuration setting sets the limit in the size of the

# set in order to use this special memory saving encoding.

set-max-intset-entries 512

# Similarly to hashes and lists, sorted sets are also specially encoded in

# order to save a lot of space. This encoding is only used when the length and

# elements of a sorted set are below the following limits:

zset-max-ziplist-entries 128

zset-max-ziplist-value 64

# Active rehashing uses 1 millisecond every 100 milliseconds of CPU time in

# order to help rehashing the main Redis hash table (the one mapping top-level

# keys to values). The hash table implementation redis uses (see dict.c)

# performs a lazy rehashing: the more operation you run into an hash table

# that is rhashing, the more rehashing "steps" are performed, so if the

# server is idle the rehashing is never complete and some more memory is used

# by the hash table.

#

# The default is to use this millisecond 10 times every second in order to

# active rehashing the main dictionaries, freeing memory when possible.

#

# If unsure:

# use "activerehashing no" if you have hard latency requirements and it is

# not a good thing in your environment that Redis can reply form time to time

# to queries with 2 milliseconds delay.

#

# use "activerehashing yes" if you don't have such hard requirements but

# want to free memory asap when possible.

activerehashing yes

################## INCLUDES ###################################

# Include one or more other config files here. This is useful if you

# have a standard template that goes to all redis server but also need

# to customize a few per-server settings. Include files can include

# other files, so use this wisely.

#

# include /path/to/local.conf

# include /path/to/other.conf