第 21 课 Redis 入门实践

1、Redis 介绍

Redis 是一种高级 key-value 数据库。它跟 memcached 类似，不过数据可以持久化，而且支持的数据类型很丰富。有字符串，链表、哈希、集合和有序集合 5 种。支持在服务器端计算集合的并、交和补集(difference)等，还支持多种排序功能。所以 Redis 也可以被看成是一个数据结构服务器。Redis 的所有数据都是保存在内存中，然后不定期的通过异步方式保存到磁盘上(这称为“半持久化模式”)；也可以把每一次数据变化都写入到一个 append only file(aof)里面(这称为“全持久化模式”)。

2、安装 redis

$ cd redis-3.2.9
$ make
$make install
$cp redis.conf /etc/

参数介绍：
make install 命令执行完成后，会在 redis-3.2.9/src 目录下生成本个可执行文件，分别是 redis-server、redis-cli、redis-benchmark、redis-check-aof 、redis-check-dump，它们的作用如下：
redis-server：Redis 服务器的 daemon 启动程序
redis-cli：Redis 命令行操作工具。也可以用 telnet 根据其纯文本协议来操作
redis-benchmark：Redis 性能测试工具，测试 Redis 在当前系统下的读写性能
redis-check-aof：数据修复
redis-check-dump：检查导出工具

3、修改系统配置文件，执行命令

a) echo vm.overcommit_memory=1 >> /etc/sysctl.conf
b) sysctl vm.overcommit_memory=1
或执行 echo vm.overcommit_memory=1 >>/proc/sys/vm/overcommit_memory

使用数字含义：
0，表示内核将检查是否有足够的可用内存供应用进程使用；如果有足够的可用内存，内存申请允许；否则，内存申请失败，并把错误返回给应用进程。
1，表示内核允许分配所有的物理内存，而不管当前的内存状态如何。

2，表示内核允许分配超过所有物理内存和交换空间总和的内存

4、修改 redis 配置文件

a) $ cd /etc
b) vi redis.conf
c) 修改 daemonize yes---目的使进程在后台运行

参数介绍：
daemonize yes：　是否以后台 daemon 方式运行。
pidfile /var/run/redis.pid：　当 Redis 以守护进程方式运行时，Redis 默认会把 pid 写入 /var/run/redis.pid 文件，可以通过 pidfile 指定。
port 6379：　指定 Redis 监听端口，默认端口为 6379。
timeout 300：　当客户端闲置多长时间后关闭连接，如果指定为 0，表示关闭该功能。
loglevel verbose：　指定日志记录级别，Redis 总共支持四个级别：debug、verbose、notice、warning，默认为 verbose。
Logfile　：日志记录方式，默认为标准输出，可指定一个文件，将日志输出到这里。
databases 16：设置开启数据库的数量，默认数据库为 0。

#　下面的例子的意思是：
#　900 秒内如果至少有 1 个 key 的值变化，则保存
#　300 秒内如果至少有 10 个 key 的值变化，则保存
#　60 秒内如果至少有 10000 个 key 的值变化，则保存
#
#　注意：也可以注释掉所有的 save 行来停用保存功能。save 表示在一定时间内执行一定数量的写操作时，自动保存快照。可设置多个条件。
#save 900 1
#save 300 10
#save 60 10000

rdbcompression yes：指定存储至本地数据库时是否压缩数据，默认为 yes，Redis 采用 LZF 压缩，如果为了节省 CPU 时间，可以关闭该选项，但会导致数据库文件变的巨大
dbfilename dump.rdb：数据快照文件名（只是文件名，不包括目录）
dir ./　数据快照的保存目录（这个是目录）
slaveof 192.168.1.100 6379　设置当本机为 slave 服务时，设置 master 服务的 IP 地址及端口，在 Redis 启动时，它会自动从 master 进行数据同步
appendonly no：是否开启 appendonlylog，开启的话每次写操作会记一条 log，这会提高数据抗风险能力，但影响效率。Redis 在默认情况下是异步的把数据写入磁盘，如果不开启，可能会在断电时导致一段时间内的数据丢失。因为 redis 本身同步数据文件是按上面 save 条件来同步的，所以有的数据会在一段时间内只存在于内存中。默认为 no。
appendfilename appendonly.aof：　指定更新日志文件名，默认为 appendonly.aof

appendfsync everysec： 指定更新日志条件，共有 3 个可选值：
no：表示等操作系统进行数据缓存同步到磁盘（快）
always：表示每次更新操作后手动调用 fsync()将数据写到磁盘（慢，安全）
everysec：表示每秒同步一次（折衷，默认值）

maxmemory 8G：指定 Redis 最大内存限制，Redis 在启动时会把数据加载到内存中，达到最大内存后，Redis 会先尝试清除已到期或即将到期的 Key，当此方法处理后，仍然到达最大内存设置，将无法再进行写入操作，但仍然可以进行读取操作。Redis 新的 vm 机制，会把 Key 存放内存，Value 会存放在 swap 区。

maxclients 128：设置同一时间最大客户端连接数，默认无限制，Redis 可以同时打开的客户端连接数为 Redis 进程可以打开的最大文件描述符数，如果设置 maxclients 0，表示不作限制。当客户端连接数到达限制时，Redis 会关闭新的连接并向客户端返回 max number of clients reached 错误信息。

5、启动 redis

a) $ cd /usr/local/bin
b) ./redis-server /etc/redis.conf

检查是否启动成功
a) $ ps -ef | grep redis

6、Redis 监控

首先判断客户端和服务器连接是否正常
客户端和服务器连接正常，返回 PONG
redis> PING
PONG
客户端和服务器连接不正常(网络不正常或服务器未能正常运行)，返回连接异常
redis 127.0.0.1:6379> PING
Could not connect to Redis at 127.0.0.1:6379: Connection refused

Redis 监控最直接的方法就是使用系统提供的 info 命令，只需要执行下面一条命令，就能获得 Redis 系统的状态报告。
redis-cli info

结果会返回 Server、Clients、Memory、Persistence、Stats、Replication、CPU、Keyspace 8 个部分。从 info 大返回结果中提取相关信息，就可以达到有效监控的目的。

先解释下各个参数含义
Server

redis_version:2.8.8 # Redis 的版本
redis_git_sha1:00000000
redis_git_dirty:0
redis_build_id:bf5d1747be5380f
redis_mode:standalone
os:Linux 2.6.32-220.7.1.el6.x86_64 x86_64
arch_bits:64
multiplexing_api:epoll
gcc_version:4.4.7 #gcc 版本
process_id:49324 # 当前 Redis 服务器进程 id
run_id:bbd7b17efcf108fdde285d8987e50392f6a38f48
tcp_port:6379
uptime_in_seconds:1739082 # 运行时间(秒)
uptime_in_days:20 # 运行时间(天)
hz:10
lru_clock:1734729
config_file:/home/s/apps/RedisMulti_video_so/conf/zzz.conf

Clients
connected_clients:1 #连接的客户端数量
client_longest_output_list:0
client_biggest_input_buf:0
blocked_clients:0

Memory
used_memory:821848 #Redis 分配的内存总量
used_memory_human:802.59K
used_memory_rss:85532672 #Redis 分配的内存总量(包括内存碎片)
used_memory_peak:178987632
used_memory_peak_human:170.70M #Redis 所用内存的高峰值
used_memory_lua:33792
mem_fragmentation_ratio:104.07 #内存碎片比率
mem_allocator:tcmalloc-2.0

Persistence
loading:0
rdb_changes_since_last_save:0 #上次保存数据库之后，执行命令的次数
rdb_bgsave_in_progress:0 #后台进行中的 save 操作的数量
rdb_last_save_time:1410848505 #最后一次成功保存的时间点，以 UNIX 时间戳格式显示
rdb_last_bgsave_status:ok
rdb_last_bgsave_time_sec:0
rdb_current_bgsave_time_sec:-1
aof_enabled:0 #redis 是否开启了 aof

```
aof_rewrite_in_progress:0
aof_rewrite_scheduled:0
aof_last_rewrite_time_sec:-1
aof_current_rewrite_time_sec:-1
aof_last_bgrewrite_status:ok
aof_last_write_status:ok

# Stats
total_connections_received:5705          #运行以来连接过的客户端的总数量
total_commands_processed:204013          # 运行以来执行过的命令的总数量
instantaneous_ops_per_sec:0
rejected_connections:0
sync_full:0
sync_partial_ok:0
sync_partial_err:0
expired_keys:34401                       #运行以来过期的 key 的数量
evicted_keys:0                           #运行以来删除过的 key 的数量
keyspace_hits:2129                       #命中 key 的次数
keyspace_misses:3148                     #没命中 key 的次数
pubsub_channels:0                        #当前使用中的频道数量
pubsub_patterns:0                        #当前使用中的模式数量
latest_fork_usec:4391

# Replication
role:master                              #当前实例的角色 master 还是 slave
connected_slaves:0
master_repl_offset:0
repl_backlog_active:0
repl_backlog_size:1048576
repl_backlog_first_byte_offset:0
repl_backlog_histlen:0

# CPU
used_cpu_sys:1551.61
used_cpu_user:1083.37
used_cpu_sys_children:2.52
used_cpu_user_children:16.79

# Keyspace
db0:keys=3,expires=0,avg_ttl=0           #各个数据库的 key 的数量，以及带有生
存期的 key 的数量
内存使用
```

如果 Redis 使用的内存超出了可用的物理内存大小，那么 Redis 很可能系统会被杀

掉。针对这一点，你可以通过 info 命令对 used_memory 和 used_memory_peak 进行监控，为使用内存量设定阀值，并设定相应的报警机制。当然，报警只是手段，重要的是你得预先计划好，当内存使用量过大后，应该做些什么，是清除一些没用的冷数据，还是把 Redis 迁移到更强大的机器上去。

持久化

如果因为你的机器或 Redis 本身的问题导致 Redis 崩溃了，那么你唯一的救命稻草可能就是 dump 出来的rdb文件了,所以,对 Redis dump 文件进行监控也是很重要的。可以通过对 rdb_last_save_time 进行监控，了解最近一次 dump 数据操作的时间，还可以通过对 rdb_changes_since_last_save 进行监控来获得如果这时候出现故障，会丢失（即已改变）多少数据。