02 Redis初识

Redis： Remote Dictionary Server 远程字典服务

为什么不叫RDS呢？因为这个名字已经被用了：Relational Database Service （关系型数据服务） 通常别用来指代MySQL。所以别出心裁，拿出了几个字母 Redis。

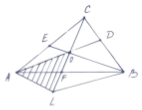
Redis是一个开源的、使用C语言编写的、支持网络交互的、可基于内存也可持久化的Key-Value数据库（这是Redis不同于Memcache的一点，支持持久化，这就让Redis更加的强大，数据也更加的安全）。

不仅仅支持简单的key-value类型的数据，同时还提供list，set，zset(sorted set)，hash 等数据结构的存储。支持数据的备份（对于数据的保障，除了持久化还有备份，如果服务本身死掉了，机器本身死掉了，那么数据不就丢失了吗，所有备份，很多情况都是用主备模式的备份master-slave），即master-slave模式的数据备份。

Redis是单进程，默认有16个数据库(计数从0开始到15)。

官网给出，每秒可以支持10万次的读写，性能非常强大。

Redis端口的由来



Redis由意大利人开发。网名Antirez。

默认端口6379取自手机键盘字母“MERZ”。

由于意大利广告女郎“Alessia Merz”在电视节目中说了一堆愚蠢的话，而被冠以“愚蠢”的代名词。

Antirez已经四十多岁，仍然孜孜不倦地为redis的开源事业做贡献。

Redis的安装

VirtualBox + Centos7
l 安装virtualBox (https : / /www.virtualbox.org/wiki /Downloads)
l 导入centos7.ova （已配置好的虚拟机镜像） - 网络是桥接网络
l 启动虚拟机，输入账号和密码(root \123456)
l 设置网络 获取ip地址 （ifconfig）
l 安装secureCRT（https : / /www.cnblogs .com/yjd_hycf _ space/p/7729796.html）
l 和虚拟机建立连接，修改窗口的编码格式 (Options - Terminal - Appearance - CharacterEncoding - UTF-8)
l 进入redis安装目录 cd /usr/local/bin
查找安装路径的方式：

方式1：

```
[root@VM-0-10-centos bin]# redis-server &
[1] 31438
[root@VM-0-10-centos bin]# 31438:C 02 Jul 13:47:14.939 # Warning: no config file specified, using
 the default config. In order to specify a config file use redis-server /path/to/redis.conf
                _.-``__ ''-._
           _.-``    `.  `_.  ''-._           Redis 3.2.12 (00000000/0) 64 bit
       .-`` .-```.  ```\/    _.,_ ''-._
      (    '      ,       .-`  | `,    )     Running in standalone mode
      |`-._`-...-` __...-.``-._|'` _.-'|     Port: 6379
      |    `-._   `._    /     _.-'    |     PID: 31438
       `-._    `-._  `-./  _.-'    _.-'
      |`-._`-._    `-.__.-'    _.-'_.-'|
      |    `-._`-._        _.-'_.-'    |           http://redis.io
       `-._    `-._`-.__.-'_.-'    _.-'
      |`-._`-._    `-.__.-'    _.-'_.-'|
      |    `-._`-._        _.-'_.-'    |
       `-._    `-._`-.__.-'_.-'    _.-'
           `-._    `-.__.-'    _.-'
               `-._        _.-'
                   `-.__.-'

31438:M 02 Jul 13:47:14.940 # Server started, Redis version 3.2.12
31438:M 02 Jul 13:47:14.940 # WARNING you have Transparent Huge Pages (THP) support enabled in yo
ur kernel. This will create latency and memory usage issues with Redis. To fix this issue run the
 command 'echo never > /sys/kernel/mm/transparent_hugepage/enabled' as root, and add it to your /
etc/rc.local in order to retain the setting after a reboot. Redis must be restarted after THP is
disabled.
31438:M 02 Jul 13:47:14.940 * The server is now ready to accept connections on port 6379

[root@VM-0-10-centos bin]# ps -ef|grep redis
root     31438 29676  0 13:47 pts/0    00:00:00 redis-server *:6379
root     31501 29676  0 13:47 pts/0    00:00:00 grep --color=auto redis
[root@VM-0-10-centos bin]# ls -l /proc/31438/cwd
lrwxrwxrwx 1 root root 0 7月    2 13:47 /proc/31438/cwd -> /usr/local/bin
[root@VM-0-10-centos bin]#
```

启动Redis

获取服务进程号

通过进程号查找路径

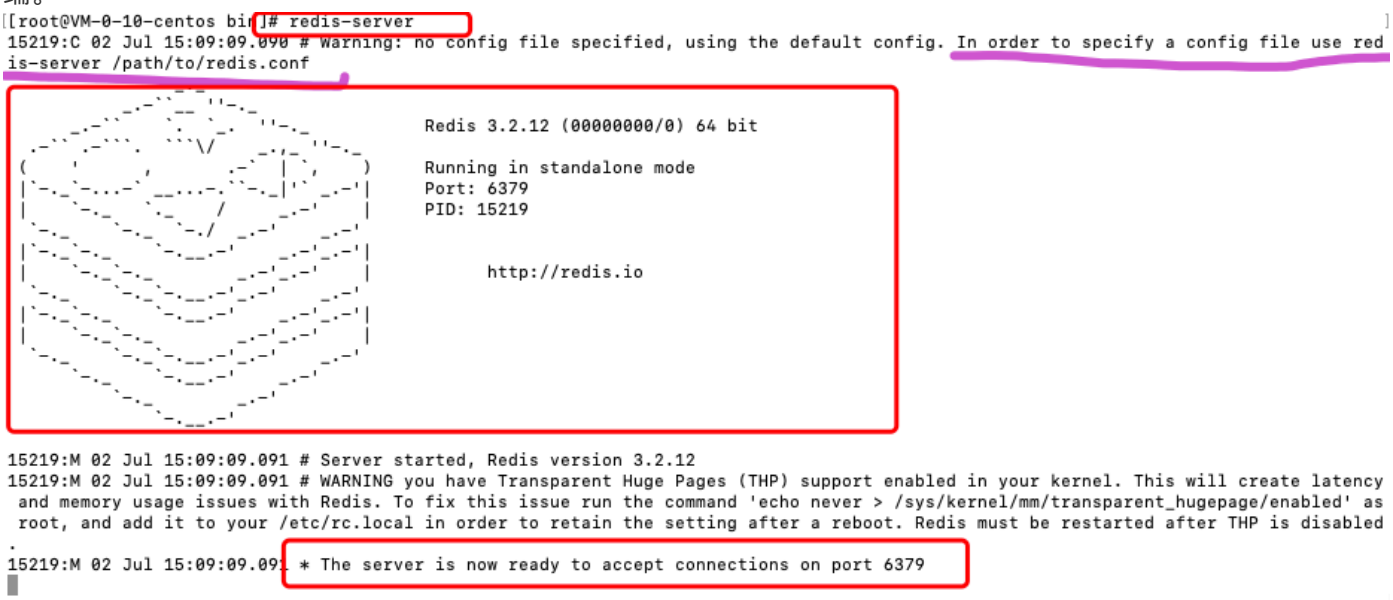
方式2：

```
[root@VM-0-10-centos bin]# # 查找Redis文件所在位置
[root@VM-0-10-centos bin]# whereis redis-server
redis-server: /usr/bin/redis-server /usr/share/man/man1/redis-server.1.gz
[root@VM-0-10-centos bin]# ll -l /usr/bin | grep redis
-rwxr-xr-x    1 root root      86944 10月  26 2018 redis-benchmark
-rwxr-xr-x    1 root root      15504 10月  26 2018 redis-check-aof
lrwxrwxrwx    1 root root         12 7月    1 15:30 redis-check-rdb -> redis-server
-rwxr-xr-x    1 root root     173448 10月  26 2018 redis-cli
lrwxrwxrwx    1 root root         12 7月    1 15:30 redis-sentinel -> redis-server
-rwxr-xr-x    1 root root     975208 10月  26 2018 redis-server
[root@VM-0-10-centos bin]#
```

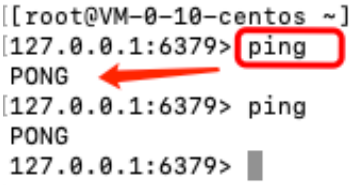
版本说明：Redis借鉴了Linux操作系统对于版本号的命名规则，版本号第二位如果是奇数，则为非稳定版本，如果是偶数，则稳定版本。

Redis的使用

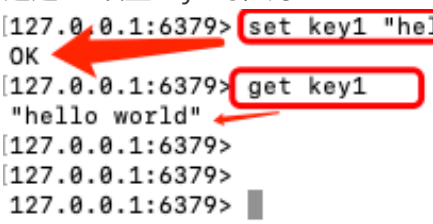
启动redis：`redis-server`，连接需要保持该窗口不关闭，clone窗口，保证IP地址不变，连接到的是当前的服务端。



客户端连接服务端：`redi-cli -p 6379`，指定端口号，并用ping去验证是否连接成功。



通过set设置key1的值为hello world，并通过get取出改key对应的值。



退出的命令：`exit`，表示的是退出该连接，实际上服务还是保持的。

也可以通过客户端，去终止服务：`shutdown`

redis-markbench，压测的命令，测试在不同命令下，10w次请求在多长时间内完成。官方标准是10w/s的get、set

```
[root@VM-0-10-centos 备份]# redis-benchmark
====== PING_INLINE ======
  100000 requests completed in 1.19 seconds
  50 parallel clients
  3 bytes payload
  keep alive: 1

99.66% <= 1 milliseconds
99.90% <= 2 milliseconds
99.96% <= 3 milliseconds
100.00% <= 3 milliseconds
84245.99 requests per second

====== PING_BULK ======
  100000 requests completed in 1.16 seconds
  50 parallel clients
  3 bytes payload
  keep alive: 1

99.79% <= 1 milliseconds
99.98% <= 2 milliseconds
100.00% <= 2 milliseconds
86058.52 requests per second

====== SET ======
  100000 requests completed in 1.21 seconds
  50 parallel clients
  3 bytes payload
  keep alive: 1

99.46% <= 1 milliseconds
99.80% <= 2 milliseconds
99.85% <= 3 milliseconds
99.95% <= 5 milliseconds
100.00% <= 5 milliseconds
82644.62 requests per second

====== GET ======
  100000 requests completed in 1.17 seconds
  50 parallel clients
  3 bytes payload
  keep alive: 1

99.87% <= 1 milliseconds
99.95% <= 2 milliseconds
100.00% <= 2 milliseconds
85251.49 requests per second

====== INCR ======
```

是不同命令的测试，不只局限于get、set

Redis配置之database 16

【注意】进入Redis的安装路径，将redis.conf先拷贝一份到自己可以找到的地方，然后再对redis.conf进行操作。

1) dataset 16

```
# Set the number of databases. The default database is DB 0, you can select
# a different one on a per-connection basis using SELECT <dbid> where
# dbid is a number between 0 and 'databases'-1
databases 16
```

默认是16个数据库，标号是0-15，默认进入的是0号数据库，可以通过select + id 切换。（我们经常是将具有相同特征的数据放在一个库中，多个数据库协同使用。）

演示：在当前库查看我们插入的key1，切换库，查看没有key。

```
127.0.0.1:6379> keys *        查看所有的key                    redis-3.2.12
1) "mylist"
2) "counter:__rand_int__"
3) "key:__rand_int__"
4) "key"        很奇怪，key1 变成了 key，难道说只有一个key的时候，默认名会更改?
127.0.0.1:6379> get key
"hello world"
127.0.0.1:6379> select 1
OK
127.0.0.1:6379[1]> keys *
(empty list or set)
127.0.0.1:6379[1]>
```

```
1. keys * 查看当前数据库所有的key值，系统会设置3个默认的key，额外使用；
2. select 1 切换到1号数据库，get key，返回nil，表示空的意思，通过127.0.0.1：6379[?]后面的?号值，可以判断当前所在数据库；
3. keys * 的性能很差，所有介绍下面这种匹配方式， keys ? 匹配，
    对于存在key1， key2， key3形式的key时，keys key? 可以找到
                                          keys k??? 也可以找到
                                          说明?是一个匹配符，并且有占位的作用。
4. dbszie 展示数据库中，key的个数。
5. 如果不想要当前数据库里面的数据了，flushdb，全部清除"慎用"。
6. flushall 删除所有数据库的数据 "要怎么用，看着办"
```

键值key操作

```
1. exists keyName : 判断keyName是否存在，存在返回（Integer）1，不存在返回（Integer）0
      * 如果keyName存在，现在执行 set keyName "value11"，获取到的值是后设置的值，前一个值被覆盖了。
2. type keyName : 显示当前键值存储的数据类型
```

到这里，我们知道，我们已经能够将数据存储起来了，那么我们要解决的关键问题是缓存，那么缓存什么最重要呢?

缓存有效期和过期策略

有效期叫做 TTL （Time to live）
设置有效期的作用：节省空间和数据弱一致性——有效期失效后保证数据的一致性

过期/淘汰策略：
当内存使用达到最大值时，需要使用某种算法来决定清理掉哪些数据，以保证新数据的存入。

FIFO

First In First Out，

先进先出。判断被存储的时间，离目前最远的数据优先被淘汰。

LRU

Least Recently Used，

最近最少使用。判断最近被使用的时间，目前最远的数据优先被淘汰。

LFU

Least Frequently Used，

最不经常使用。在一段时间内，数据被使用次数最少的，优先被淘汰。

```
3. expire keyName second : 设置keyName的有效期为second秒。
```

```
   ttl keyName : 查看keyName是否过期，返回（Integer）n .
      如果n>0，代表未过期，n此时表示剩余过期的秒数；
      如果n=-2，代表这个keyName已经过期了，且keyName已经不存在；
      如果n=-1，代表没有设置keyName有效期。
4.persist keyName : 设置数据一直有效，特别作用于错误设置了有效期后。
      如果返回（Integer）0，表示，操作失败（原因可能在于该keyName，已经不存在了）
      如果返回（Integer）1，表示，操作成功。
5. del keyName : 删除keyName，返回（Integer）1，表示成功；返回（Integer）0，表示失败。
6. rename keyName1 keyName2 : 将keyName1的名字变更为keyName2，返回OK，表示成功。
7. randomkey : 随机返回一个key。
8. move keyName dbid : 移动指定key到对应的dbid数据库中。
9. expire 和 ttl操作的都是秒级的，但是有时候我们需要精细化我们的时间，使用毫秒级，
       pexpire keyName milSecond : 设置keyName的有效期为milSecond;
       pttl keyName : 查看剩余有效期的毫秒数
```