redis

2025.1.19

1.安装

redis安装后的一些文件默认是在/usr/local/bin目录下的：

```
总用量 21492
-rwxr-xr-x. 1 root root  5197848 12月  10 19:47 redis-benchmark
lrwxrwxrwx. 1 root root       12 12月  10 19:47 redis-check-aof -> redis-server
lrwxrwxrwx. 1 root root       12 12月  10 19:47 redis-check-rdb -> redis-server
-rwxr-xr-x. 1 root root  5411040 12月  10 19:47 redis-cli
lrwxrwxrwx. 1 root root       12 12月  10 19:47 redis-sentinel -> redis-server
-rwxr-xr-x. 1 root root 11390192 12月  10 19:47 redis-server
```

自上而下分别是redis的测试工具、修复有问题的aof文件、修复有问题的rdb文件、客户端操作入口、Redis集群哨兵、redis服务器启动命令。

并且自带一个出厂默认的配置文件（/export/server/redis）：

```
[root@centos redis]# ll
总用量 264
-rw-rw-r--.  1 root root  34885 7月   18 2022 00-RELEASENOTES
-rw-rw-r--.  1 root root     51 7月   18 2022 BUGS
-rw-rw-r--.  1 root root   5027 7月   18 2022 CODE_OF_CONDUCT.md
-rw-rw-r--.  1 root root   2634 7月   18 2022 CONTRIBUTING.md
-rw-rw-r--.  1 root root   1487 7月   18 2022 COPYING
drwxrwxr-x.  7 root root    187 12月  10 19:41 deps
-rw-rw-r--.  1 root root     11 7月   18 2022 INSTALL
-rw-rw-r--.  1 root root    151 7月   18 2022 Makefile
-rw-rw-r--.  1 root root   6888 7月   18 2022 MANIFESTO
-rw-rw-r--.  1 root root  22441 7月   18 2022 README.md
-rw-rw-r--.  1 root root 106545 7月   18 2022 redis.conf
-rwxrwxr-x.  1 root root    279 7月   18 2022 runtest
-rwxrwxr-x.  1 root root    283 7月   18 2022 runtest-cluster
-rwxrwxr-x.  1 root root   1578 7月   18 2022 runtest-moduleapi
-rwxrwxr-x.  1 root root    285 7月   18 2022 runtest-sentinel
-rw-rw-r--.  1 root root   1695 7月   18 2022 SECURITY.md
-rw-rw-r--.  1 root root  14005 7月   18 2022 sentinel.conf
drwxrwxr-x.  4 root root  12288 12月  10 19:47 src
drwxrwxr-x. 11 root root    199 7月   18 2022 tests
-rw-rw-r--.  1 root root   3055 7月   18 2022 TLS.md
drwxrwxr-x.  8 root root   4096 7月   18 2022 utils
```

2025.1.20

2.怎么玩?

1.修改配置文件

redis的运行依赖于自定义的配置文件，这类配置文件在出厂时默认会自带一个，我们一般把默认的配置文件进行复制，就是保留一个备份，这是个好习惯，这里我们保存在/export/server/myredis下：

```
[root@centos myredis]# ll
总用量 108
-rw-r--r--. 1 root root 106545 1月  19 10:27 redis.conf
```

然后修改我们的配置文件，修改后记得重启，因为不是实时生效

使用vim进入配置文件：

```
# Redis configuration file example.
#
# Note that in order to read the configuration file, Redis must be
# started with the file path as first argument:
#
# ./redis-server /path/to/redis.conf

# Note on units: when memory size is needed, it is possible to specify
# it in the usual form of 1k 5GB 4M and so forth:
#
# 1k => 1000 bytes
# 1kb => 1024 bytes
# 1m => 1000000 bytes
# 1mb => 1024*1024 bytes
# 1g => 1000000000 bytes
# 1gb => 1024*1024*1024 bytes
#
# units are case insensitive so 1GB 1Gb 1gB are all the same.

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

# Include one or more other config files here.  This is useful if you
# have a standard template that goes to all Redis servers but also need
# to customize a few per-server settings.  Include files can include
# other files, so use this wisely.
#
# Note that option "include" won't be rewritten by command "CONFIG REWRITE"
# from admin or Redis Sentinel. Since Redis always uses the last processed
# line as value of a configuration directive, you'd better put includes
# at the beginning of this file to avoid overwriting config change at runtime.
#
```

使用set nu显示行号

1.修改daemonize为true（如果是docker则不能设置为true）

使用/daemonize查找：

```
################################ GENERAL #####################################

# By default Redis does not run as a daemon. Use 'yes' if you need it.
# Note that Redis will write a pid file in /var/run/redis.pid when daemonized.
# When Redis is supervised by upstart or systemd, this parameter has no impact.
daemonize no
```

这个是干什么的呢？我们尝试使用redis-server启动一下redis就会发现，虽然redis启动了，但是是以前台方式启动的，也就是说我们在启动后无法进行其他输入操作，画面会停在redis的logo界面，作为一个后端服务器我们希望他以后台方式启动，所以将其改为yes：

```
304 ################################ GENERAL #####################################
305
306 # By default Redis does not run as a daemon. Use 'yes' if you need it.
307 # Note that Redis will write a pid file in /var/run/redis.pid when daemonized.
308 # When Redis is supervised by upstart or systemd, this parameter has no impact.
309 daemonize yes
310
```

2.修改protected-mode yes为protected-mode no

```
101 # Protected mode is a layer of security protection, in order to avoid that
102 # Redis instances left open on the internet are accessed and exploited.
103 #
104 # When protected mode is on and the default user has no password, the server
105 # only accepts local connections from the IPv4 address (127.0.0.1), IPv6 address
106 # (::1) or Unix domain sockets.
107 #
108 # By default protected mode is enabled. You should disable it only if
109 # you are sure you want clients from other hosts to connect to Redis
110 # even if no authentication is configured.
111 protected-mode yes
```

```
101 # Protected mode is a layer of security protection, in order to avoid that
102 # Redis instances left open on the internet are accessed and exploited.
103 #
104 # When protected mode is on and the default user has no password, the server
105 # only accepts local connections from the IPv4 address (127.0.0.1), IPv6 address
106 # (::1) or Unix domain sockets.
107 #
108 # By default protected mode is enabled. You should disable it only if
109 # you are sure you want clients from other hosts to connect to Redis
110 # even if no authentication is configured.
111 protected-mode no
```

这个是保护模式的开关，在生产模式中不建议关闭，在开发模式中可以关闭，因为我们在开发中需要让我们的springboot或者springcloud微服务连接到我们的redis，所以需要把这个关闭。

3.修改我们的bind

先找到我们的bind：

```
73 # ~~~ WARNING ~~~ If the computer running Redis is directly exposed to the
74 # internet, binding to all the interfaces is dangerous and will expose the
75 # instance to everybody on the internet. So by default we uncomment the
76 # following bind directive, that will force Redis to listen only on the
77 # IPv4 and IPv6 (if available) loopback interface addresses (this means Redis
78 # will only be able to accept client connections from the same host that it is
79 # running on).
80 #
81 # IF YOU ARE SURE YOU WANT YOUR INSTANCE TO LISTEN TO ALL THE INTERFACES
82 # COMMENT OUT THE FOLLOWING LINE.
83 #
84 # You will also need to set a password unless you explicitly disable protected
85 # mode.
86 # ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
87 bind 127.0.0.1 -::1
```

这个默认是只有127.0.0.1，也就是说只有本机才能访问我们的redis，这个在以后做微服务的时候是扩展性不高的，所以直接注释掉，让所有的ip都可以连接进我们的redis中。

```
87 #bind 127.0.0.1 -::1
```

4.添加redis密码

查找我们的requirepass：

```
1027 # IMPORTANT NOTE: starting with Redis 6 "requirepass" is just a compatibility
1028 # layer on top of the new ACL system. The option effect will be just setting
1029 # the password for the default user. Clients will still authenticate using
1030 # AUTH <password> as usually, or more explicitly with AUTH default <password>
1031 # if they follow the new protocol: both will work.
1032 #
1033 # The requirepass is not compatible with aclfile option and the ACL LOAD
1034 # command, these will cause requirepass to be ignored.
1035 #
1036 # requirepass foobared
```

这里我们设置我们的redis访问密码为111

```
1037 requirepass 111
```

2.启动redis

使用redis-server 配置文件路径　来启动我们的redis：

```
[root@centos myredis]# redis-server redis7.conf
```

查看我们的redis的运行状态：

```
[root@centos myredis]# ps -ef|grep redis|grep -v grep
root       57617       1  0 16:31 ?        00:00:00 redis-server *:6379
```

可以看到我们的redis在我们的6379端口运行

3.以客户端模式连接我们的reids

使用reids-cli -a 密码 -p 端口号　来连接到我们对应端口号的redis服务器（如果不写-p就默认连接到6379）

如果我们不写-a，我们也可以连接到redis-cli中，但是我们不能进行操作，需要使用

```
[root@centos myredis]# redis-cli -a 111 -p 6379
Warning: Using a password with '-a' or '-u' option on the command line interface may not be safe.
127.0.0.1:6379>
```

然后我们新开一个窗口查看redis的运行状况：

```
[root@centos ~]# ps -ef|grep redis
root       57617       1  0 16:31 ?        00:00:01 redis-server *:6379
root       64366    2755  0 16:36 pts/1    00:00:00 redis-cli -a 111 -p 6379
root       66026   65172  0 16:37 pts/3    00:00:00 grep --color=auto redis
[root@centos ~]#
```

可以发现除了我们的reids-server服务端外，又多了一个redis-cli客户端

使用quit命令可以退出我们的reids客户端，但仅仅是退出，不会关闭我们的redis服务器。

关闭redis服务器可以用redis-cli -a 密码 shutdown（单实例关闭）

redis-cli -p 端口号 shutdown（多实例关闭）

3.一切始于helloworld

我们连接进redis-cli后：

使用redis的基础命令：set和get：

```
127.0.0.1:6379> set k1 helloworld
OK
127.0.0.1:6379> get k1
"helloworld"
```

4.redis十大数据类型

对于数据类型的操作命令，可以去Commands | Docs查找，也可以问ai

首先是对于redis中的key操作：

keys *-查找所有的key

exists k1-查看k1是否存在

type k1-查看k1的数据类型

del k1-删除k1，成功返回1，失败返回0

unlink k1-非阻塞删除k1，先给k1打上标记，将k1与v1的连接切断，v1真正的删除工作由后续异步操作进行

ttl k1-查看当前k1还有多少秒过期，-1为永不过期，-2为已过期

expire k1 秒数-设置k1的过期时间

move k1 dbindex [0-15]-将当前的k1移动到数据库给定的db当中

这个是什么意思呢？我们的redis数据库一共有16个库，我们一般默认的直接访问就是0号库，假如我们在0号库有k1：

```
127.0.0.1:6379> get k1
"helloworld"
```

然后我们将其移动到3号库：

```
127.0.0.1:6379> move k1 3
(integer) 1
```

我们在0号库进行查询发现没有了：

```
127.0.0.1:6379> get k1
(nil)
```

我们切换到3号库：

```
127.0.0.1:6379> select 3
OK
```

查询：

```
127.0.0.1:6379[3]> get k1
"helloworld"
```

关于数据库的操作：

使用select dbindex来切换到对应编号对的数据库，数据库一共有16个，这是由配置文件来决定的，我们进入redis的配置文件，然后查询database：

```
# 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
```

可以发现默认就是16个数据库。

dbsize-查看数据库的key的数量

flushdb-清空当前库

flushall-通杀全部库

1.String

String是redis最基本的数据类型，一个key对应一个value

String是二进制安全的，意味着String可以储存任何数据，包括jpg图片或者被序列化的对象

一个value最大为512M

2.List

3.Hash

4.集合Set

5.有序集合ZSet

6.地理空间GEO

7.基数统计HyperLogLog

8.位图bitmap

9.位域bitfield

10.流Stream