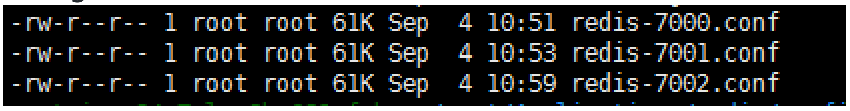
配置主从节点

此处就以端口号7000为主节点，7001为从节点，7002为从节点演示。

1.在redis目录下新建config目录，data目录

执行命令`cp redis.conf config/`,将redis.conf复制到config目录下，再执行命令
`cp redis.conf redis-7000.conf`
`cp redis.conf redis-7001.conf`
`cp redis.conf redis-7002.conf`
`rm -rf redis.conf`
config目录下就会如下图所示：



```
-rw-r--r-- 1 root root 61K Sep  4 10:51 redis-7000.conf
-rw-r--r-- 1 root root 61K Sep  4 10:53 redis-7001.conf
-rw-r--r-- 1 root root 61K Sep  4 10:59 redis-7002.conf
```

2.依此编辑三个文件并保存

`vim redis-7000.conf`

修改命令如下：

```
port 7000
daemonize yes
pidfile /var/run/redis_7000.pid
logfile "7000.log"
dbfilename dump-7000.rdb
dir ../data
```

`vim redis-7001.conf`

修改命令如下：

```
port 7000
daemonize yes
pidfile /var/run/redis_7000.pid
logfile "7000.log"
dbfilename dump-7000.rdb
dir ../data
slaveof 127.0.0.1 7000
```

`vim redis-7002.conf`

修改命令如下：

```
port 7000
daemonize yes
pidfile /var/run/redis_7000.pid
```

```
logfile "7000.log"
dbfilename dump-7000.rdb
dir ../data
slaveof 127.0.0.1 7000
```

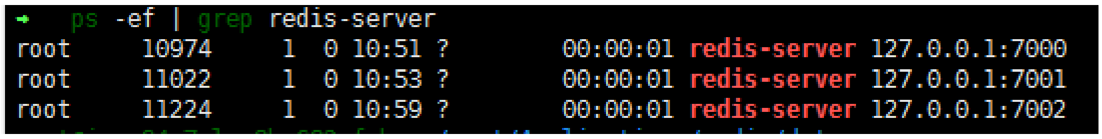
3.依此启动redis服务

```
redis-server redis-7000.conf
redis-server redis-7001.conf
redis-server redis-7002.conf
```

4.查看redis服务进程

```
ps -ef | grep redis-server
```

如下图所示，则服务启动成功



```
➜  ps -ef | grep redis-server
root     10974     1  0 10:51 ?        00:00:01 redis-server 127.0.0.1:7000
root     11022     1  0 10:53 ?        00:00:01 redis-server 127.0.0.1:7001
root     11224     1  0 10:59 ?        00:00:01 redis-server 127.0.0.1:7002
```

redis Sentinel (哨兵)

1.故障转移

1.多个sentinel发现并确定master有问题。
2.选举出一个sentinel作为领导。
3.选出一个slave作为master.
4.通知其余slave称为新的master的slave.
5.通知客户端主从变化。
6.等待老的master复活成为新的master的slave.

2.配置sentinel

1.在redis目录下执行`cp sentinel.conf config/`，
将sentinel.conf复制到config目录下，然后再执行命令
`cp sentinel.conf redis-sentinel-26379.conf`
`rm -rf sentinel.conf`

2.然后执行命令`redis-sentinel-26379.conf`

修改配置如下：

```
port 26379
daemonize yes
pidfile /var/run/redis-sentinel-26379.pid
logfile "26379.log"
dir ../data
sentinel monitor mymaster 127.0.0.1 7000 2
sentinel down-after-milliseconds mymaster 30000
sentinel parallel-syncs mymaster 1
sentinel failover-timeout mymaster 180000
sentinel deny-scripts-reconfig yes
```

3.复制redis-sentinel-26379.conf

```
sed "s/26379/26380/g" redis-sentinel-26379.conf >redis-sentinel-26380.conf
sed "s/26379/26381/g" redis-sentinel-26379.conf >redis-sentinel-26381.conf
```

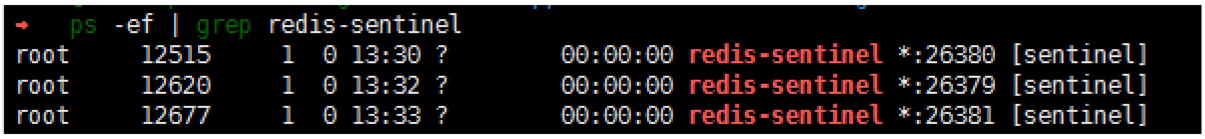
4.启动sentinel服务，依此执行如下命令：

```
redis-sentinel redis-sentinel-26379.conf
redis-sentinel redis-sentinel-26380.conf
redis-sentinel redis-sentinel-26381.conf
```

5.查看服务进程：

```
ps -ef | grep redis-sentinel
```

出现如下图所示，则代表启动成功



```
➜  ps -ef | grep redis-sentinel
root      12515     1  0 13:30 ?        00:00:00 redis-sentinel *:26380 [sentinel]
root      12620     1  0 13:32 ?        00:00:00 redis-sentinel *:26379 [sentinel]
root      12677     1  0 13:33 ?        00:00:00 redis-sentinel *:26381 [sentinel]
```

redis Cluster（集群）

计入集群的作用：

1.为它迁移槽和数据实现扩容
2.作为从节点负责故障转移

配置 redis cluster

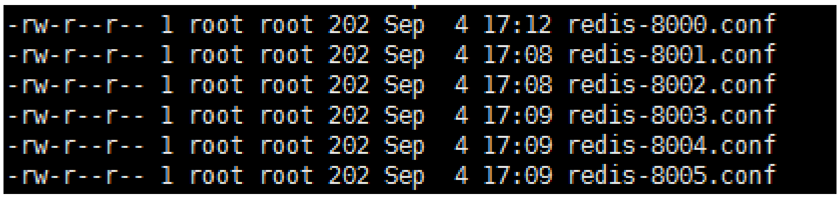
1.在redis/config目录下执行命令`vim redis-8000.conf`创建redis-8000.conf文件，在文件里编辑

```
port 8000
daemonize yes
pidfile /var/run/redis_8000.pid
logfile "8000.log"
dbfilename dump-8000.rdb
dir ../data
cluster-enabled yes
cluster-config-file nodes-8000.conf
cluster-require-full-coverage no
```

2.再执行命令复制5个文件

```
sed "s/8000/8001/g" redis-8000.conf > redis-8001.conf
sed "s/8000/8002/g" redis-8000.conf > redis-8002.conf
sed "s/8000/8003/g" redis-8000.conf > redis-8003.conf
sed "s/8000/8004/g" redis-8000.conf > redis-8004.conf
sed "s/8000/8005/g" redis-8000.conf > redis-8005.conf
```

则在config目录下出现如下所示：

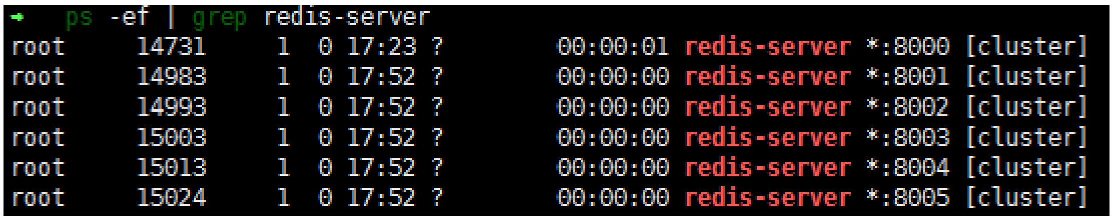


```
-rw-r--r-- 1 root root 202 Sep  4 17:12 redis-8000.conf
-rw-r--r-- 1 root root 202 Sep  4 17:08 redis-8001.conf
-rw-r--r-- 1 root root 202 Sep  4 17:08 redis-8002.conf
-rw-r--r-- 1 root root 202 Sep  4 17:09 redis-8003.conf
-rw-r--r-- 1 root root 202 Sep  4 17:09 redis-8004.conf
-rw-r--r-- 1 root root 202 Sep  4 17:09 redis-8005.conf
```

3.依此启动服务
`redis-server redis-8000.conf`

4.查看服务进程
`ps -ef | grep redis-server`

如下图所示则代表服务启动成功

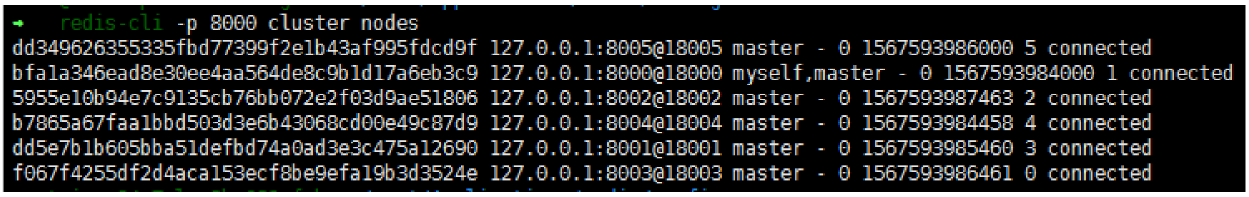


```
➜  ps -ef | grep redis-server
root     14731     1  0 17:23 ?        00:00:01 redis-server *:8000 [cluster]
root     14983     1  0 17:52 ?        00:00:00 redis-server *:8001 [cluster]
root     14993     1  0 17:52 ?        00:00:00 redis-server *:8002 [cluster]
root     15003     1  0 17:52 ?        00:00:00 redis-server *:8003 [cluster]
root     15013     1  0 17:52 ?        00:00:00 redis-server *:8004 [cluster]
root     15024     1  0 17:52 ?        00:00:00 redis-server *:8005 [cluster]
```

5.节点握手

```
redis-cli -p 8000 cluster meet 127.0.0.1 8001
redis-cli -p 8000 cluster meet 127.0.0.1 8002
redis-cli -p 8000 cluster meet 127.0.0.1 8003
redis-cli -p 8000 cluster meet 127.0.0.1 8004
redis-cli -p 8000 cluster meet 127.0.0.1 8005
```

然后执行命令`redis-cli -p 8000 cluster nodes`,显示如下：



```
➜  redis-cli -p 8000 cluster nodes
dd349626355335fbd77399f2e1b43af995fdcd9f 127.0.0.1:8005@18005 master - 0 1567593986000 5 connected
bfa1a346ead8e30ee4aa564de8c9b1d17a6eb3c9 127.0.0.1:8000@18000 myself,master - 0 1567593984000 1 connected
5955e10b94e7c9135cb76bb072e2f03d9ae51806 127.0.0.1:8002@18002 master - 0 1567593987463 2 connected
b7865a67faa1bbd503d3e6b43068cd00e49c87d9 127.0.0.1:8004@18004 master - 0 1567593984458 4 connected
dd5e7b1b605bba51defbd74a0ad3e3c475a12690 127.0.0.1:8001@18001 master - 0 1567593985460 3 connected
f067f4255df2d4aca153ecf8be9efa19b3d3524e 127.0.0.1:8003@18003 master - 0 1567593986461 0 connected
```