<https://www.jianshu.com/p/8045b92fafb2>

**注意：java程序或远程客户端中访问，需在防火墙中开放对应的端口**

**集群搭建：**

**第一步：在第一台机器的/usr/local下创建文件夹redis-cluster，然后在其下面创建6个文件夾如下:**

[root@localhost redis-5.0.5]# mkdir -p /usr/local/redis-cluster

[root@localhost redis-5.0.5]# cd /usr/local/redis-cluster/

[root@localhost redis-cluster]# mkdir 8001 8002 8003 8004 8005 8006

**第二步：把之前的redis.conf配置文件copy到8001下**

[root@localhost redis-cluster]# cp -r /usr/local/redis-5.0.5/redis.conf ./8001/

修改如下内容：

1）daemonize yes

2）port 8001（分别对每个机器的端口号进行设置）

3）dir /usr/local/redis-cluster/8001/（指定数据文件存放位置，必须要指定不同的目录位置，不然会丢失数据）

4）cluster-enabled yes（启动集群模式）

5）cluster-config-file nodes-8001.conf（集群节点信息文件，这里800x最好和port对应上）

6）cluster-node-timeout 5000

1. bind 127.0.0.1（去掉bind绑定访问ip信息）
2. protected-mode no （关闭保护模式）

9）appendonly yes

[root@localhost 8001]# vim redis.conf

daemonize yes



port 8001



dir /usr/local/redis-cluster/8002/



protected-mode no



cluster-enabled yes



cluster-config-file nodes-8001.conf



cluster-node-timeout 15000





appendonly yes



requirepass zhangjun



masterauth zhangjun





日志也可以打开：







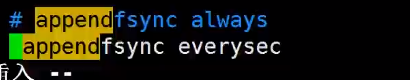


















第三步：把修改后的配置文件，copy到8002-8006，分别修改第2、3、5项里的端口号

[root@localhost 8001]# cp redis.conf ../8002/

[root@localhost 8001]# cp redis.conf ../8003/

[root@localhost 8001]# cp redis.conf ../8004/

[root@localhost 8001]# cp redis.conf ../8005/

[root@localhost 8001]# cp redis.conf ../8006/

[root@localhost 8001]# vim ../8002/redis.conf



[root@localhost 8001]# vim ../8003/redis.conf

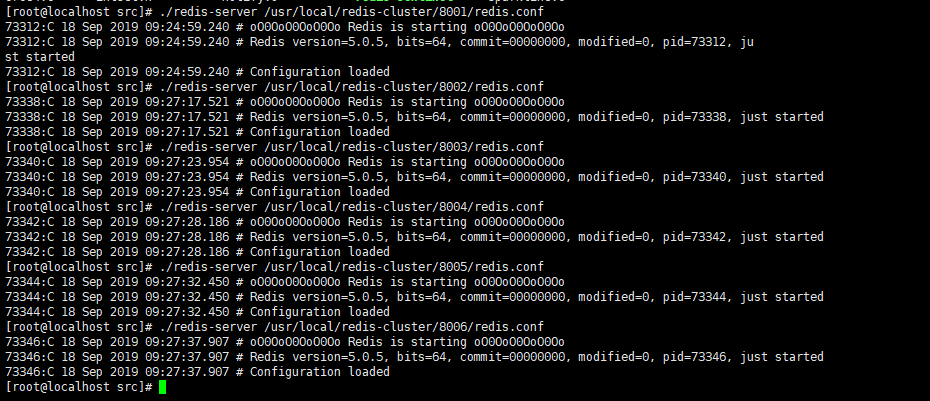
[root@localhost 8001]# vim ../8004/redis.conf

[root@localhost 8001]# vim ../8005/redis.conf

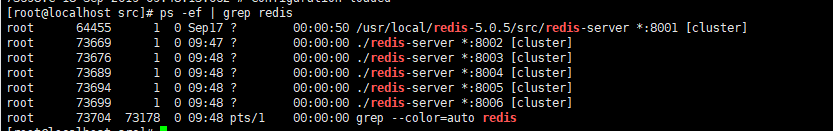
[root@localhost 8001]# vim ../8006/redis.conf

**第四步：分别启动6个redis实例，然后检查是否启动成功**

/usr/local/redis-5.0.5/src/redis-server /usr/local/redis-cluster/8001/redis.conf

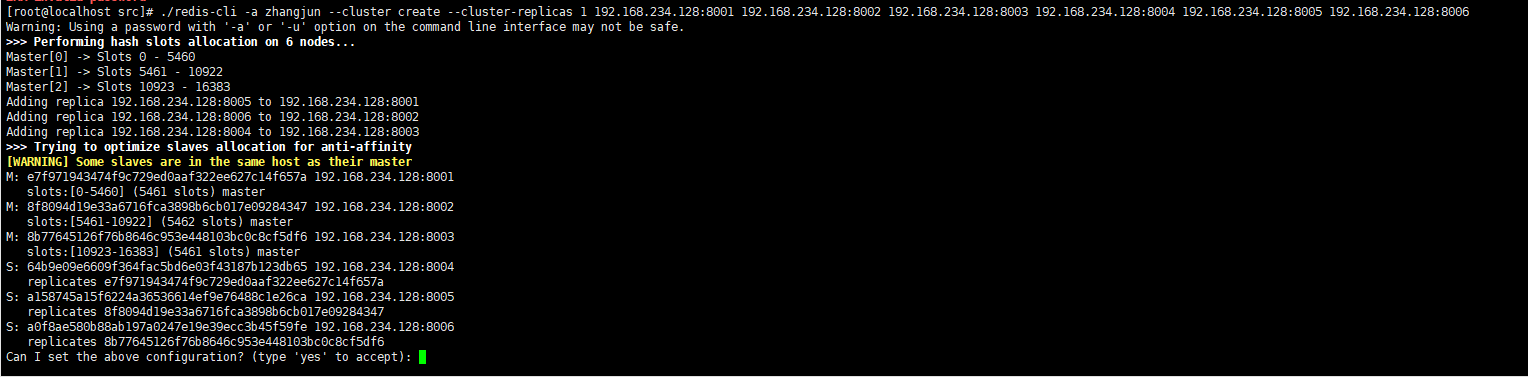


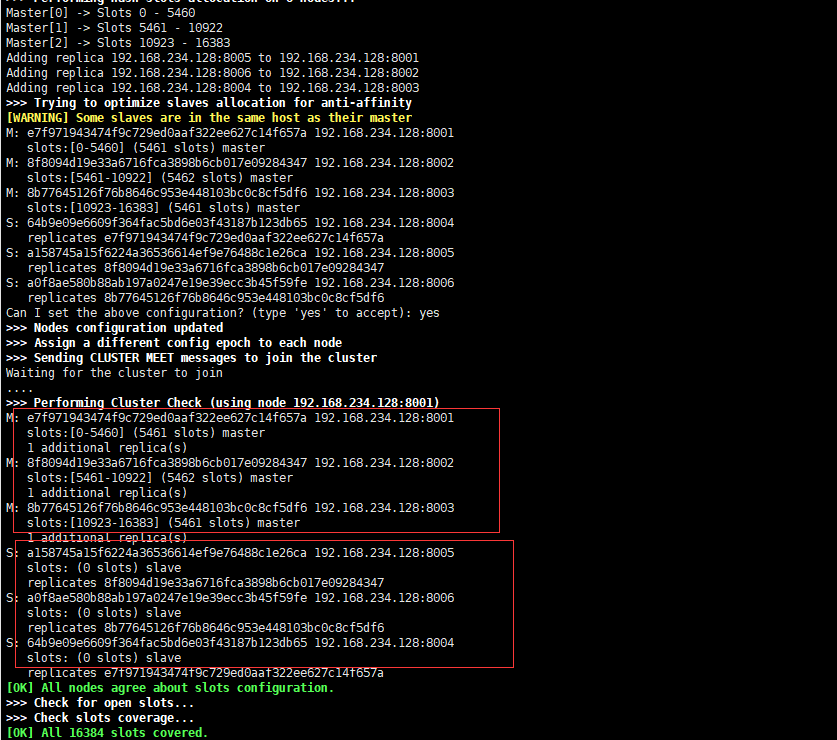
查看是否启动成功



**第五步：用redis-cli创建整个redis集群(redis5以前的版本集群是依靠ruby脚本redis-trib.rb实现),创建集群后，下一次重启服务后，集群会自动构建**

[root@localhost src]# ./redis-cli -a zhangjun --cluster create --cluster-replicas 1 192.168.234.128:8001 192.168.234.128:8002 192.168.234.128:8003 192.168.234.128:8004 192.168.234.128:8005 192.168.234.128:8006





代表为每个创建的主服务器节点创建一个从服务器节点

第七步：验证集群：

1. 连接任意一个客户端即可

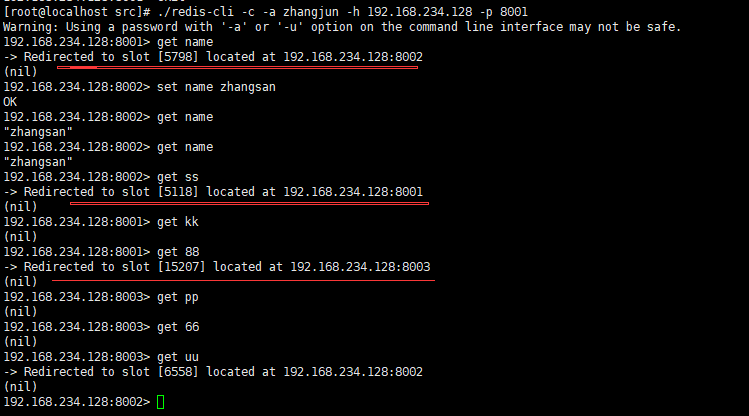
-a访问服务端密码，-c表示集群模式，指定ip地址和端口号

[root@localhost src]# ./redis-cli -c -a zhangjun -h 192.168.234.128 -p 8001

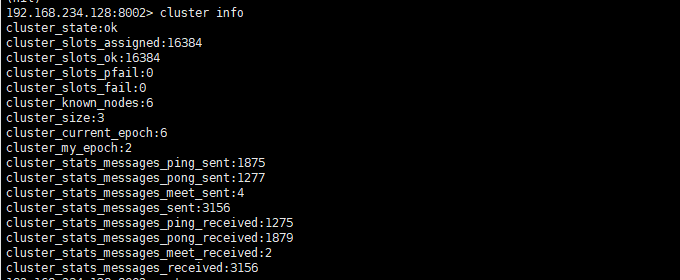


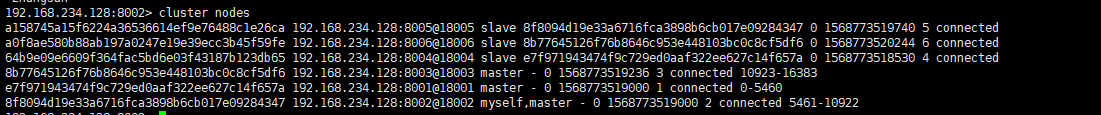
-a访问服务端密码，-c表示集群模式，指定ip地址和端口号

进入到8002了，redirected



1. 进行验证： cluster info（查看集群信息）、cluster nodes（查看节点列表）

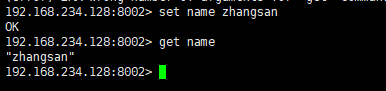


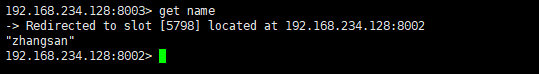


3）进行数据操作验证

redis的cluster集群，主节点提供读写操作，从节点作为备用节点，不提供请求，只作为故障转移使用。

主库8002：





从库8005：





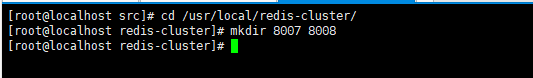
**集群动态扩容：动态添加或者删除集群中的节点**

刚刚加入集群的为master,但无槽点，需要分配

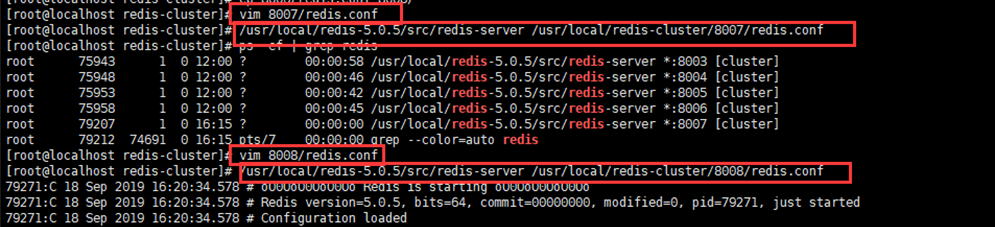
只有主节点有槽点，用于存储数据

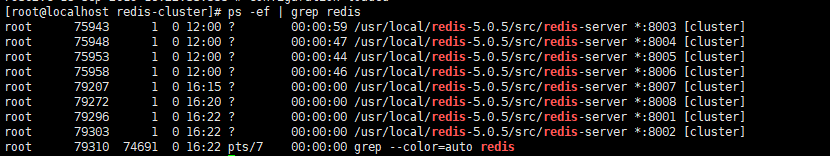
第一步：再创建两个文件夹 8007，8008

[root@localhost redis-cluster]# mkdir 8007 8008



第二步：编辑配置文件并开启实例

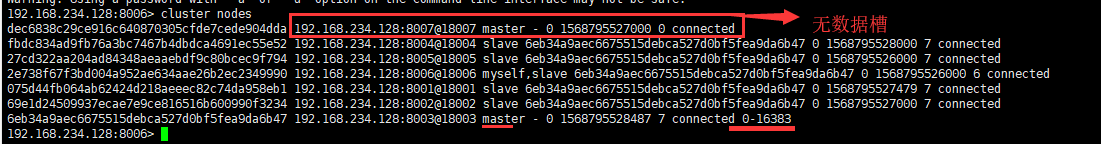




第三步：将新的两个实例加入已有集群

1、添加主节点8007进入集群，添加之后默认为主节点

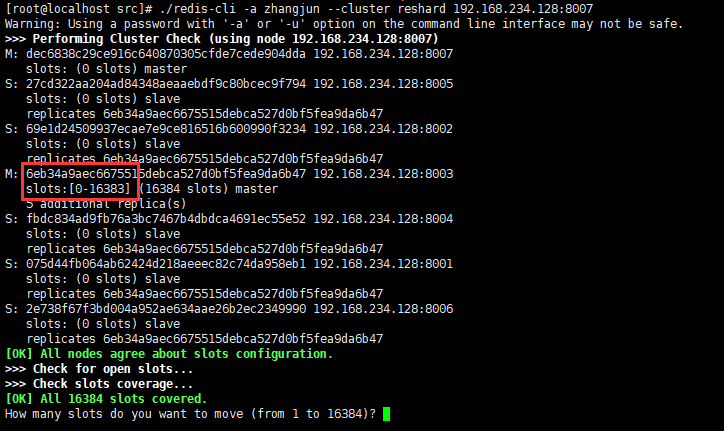
[root@localhost src]# ./redis-cli -a zhangjun --cluster add-node 192.168.234.128:8007 192.168.234.128:8006

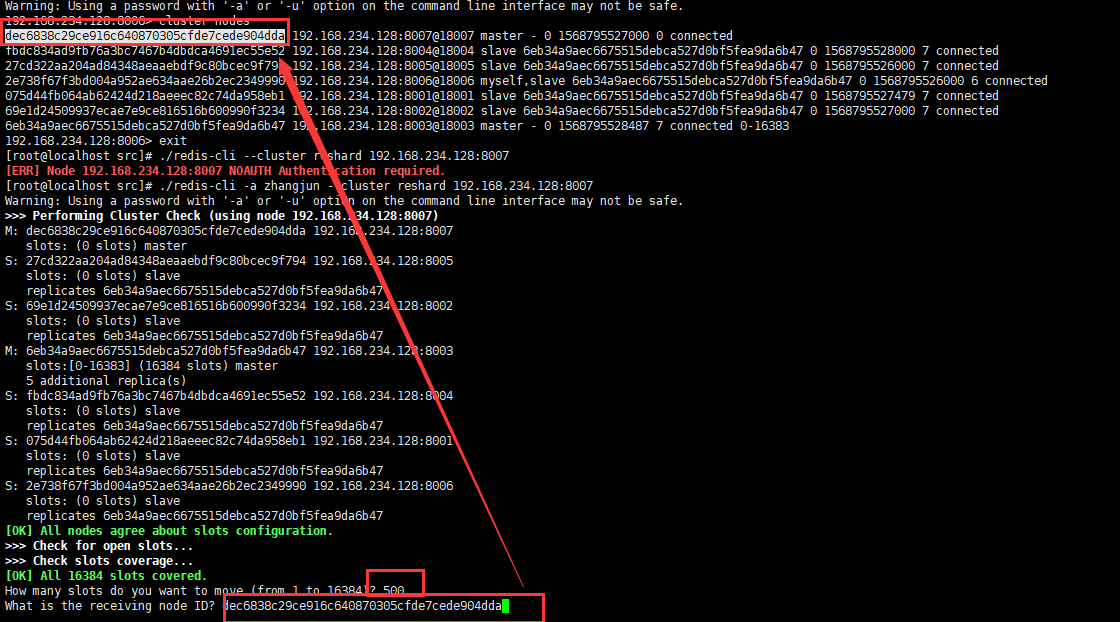


分片，给主节点8007添加槽点

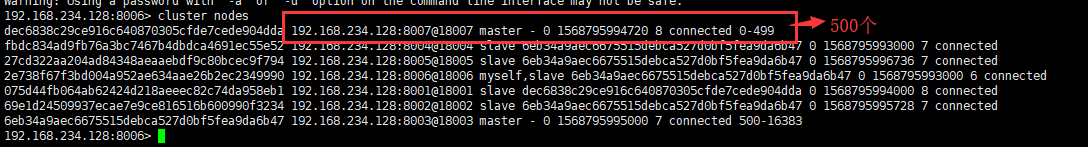
[root@localhost src]# ./redis-cli -a zhangjun --cluster reshard 192.168.234.128:8007

执行后，提示需要从别的主节点上分配多少槽点 例子：500

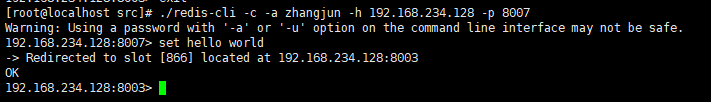


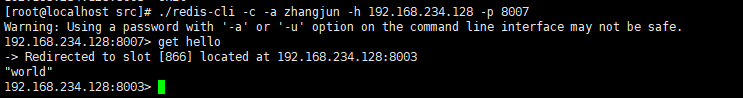


执行成功后，查看有500个



测试8007存取数据：

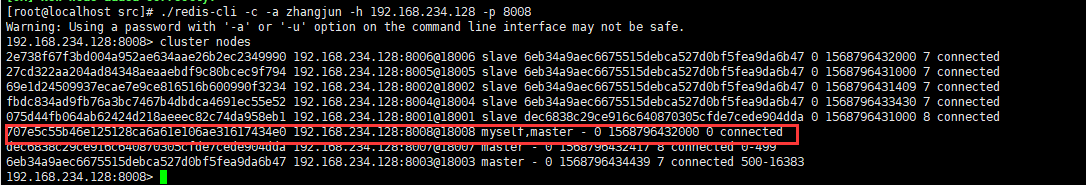




2、添加从节点8008，成为8007的从节点

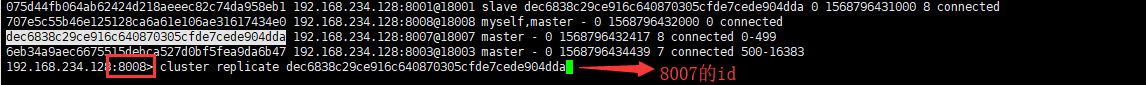
添加主节点8008进入集群，添加之后默认为主节点

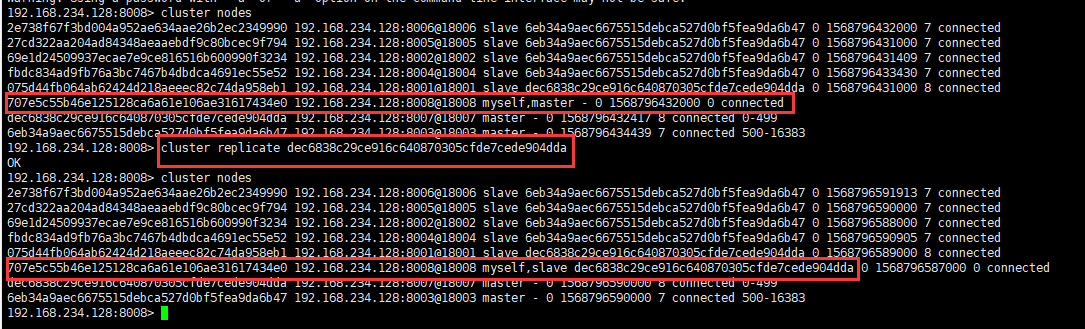
[root@localhost src]# ./redis-cli -a zhangjun --cluster add-node 192.168.234.128:8008 192.168.234.128:8007



让8008成为8007的从节点

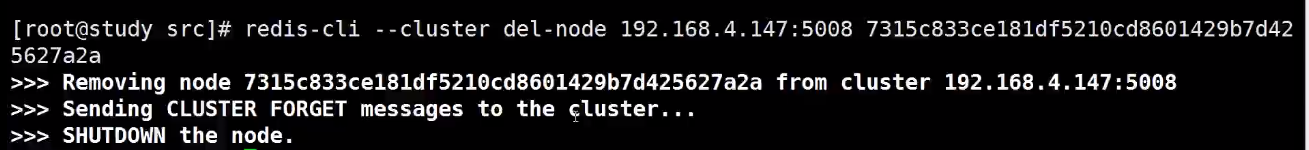
192.168.234.128:8008> cluster replicate dec6838c29ce916c640870305cfde7cede904dda





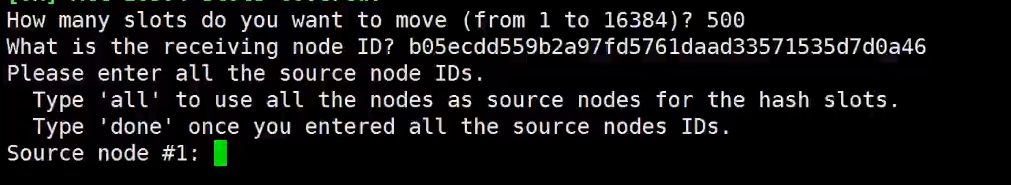
3、删除节点：

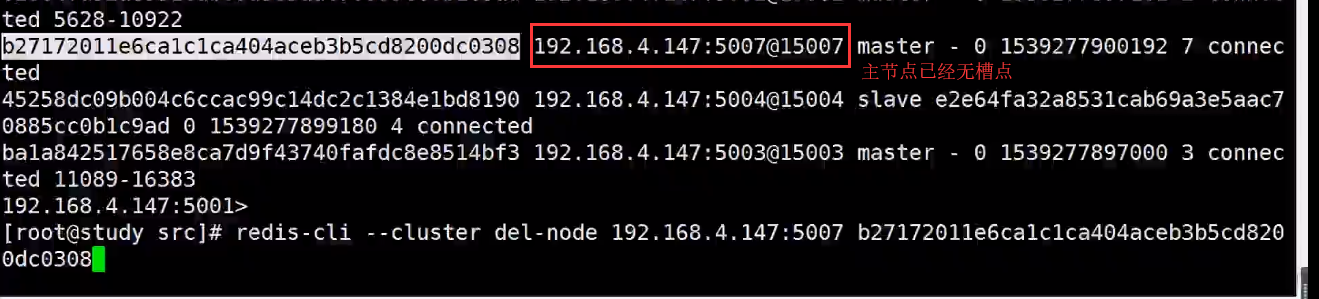
删除从节点



删除主节点，需要在删除前分片，归还数据槽







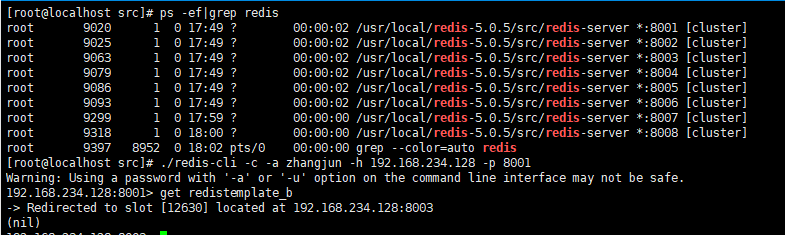
关闭集群则需要逐个进行关闭，使用命令

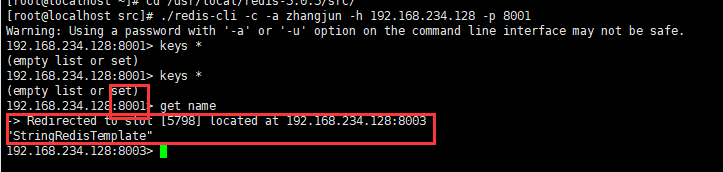
[root@localhost src]# ./redis-cli -c -a zhangjun -h 192.168.234.128 -p 8001 shutdown

查看集群信息

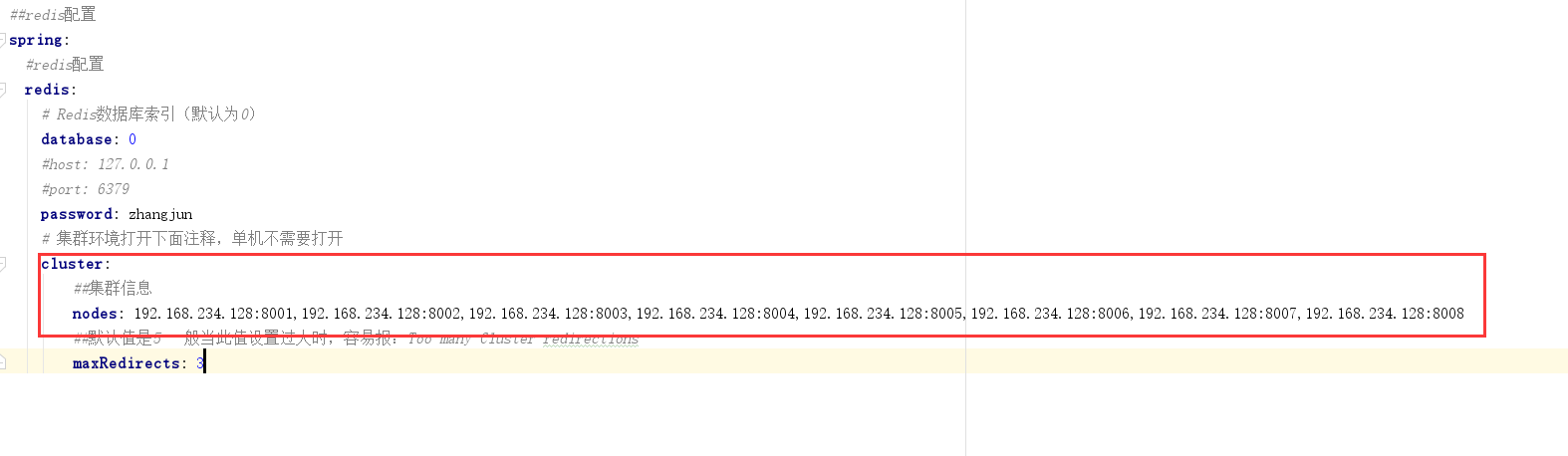
[root@localhost src]# ./redis-cli -c -a zhangjun -h 192.168.234.128 -p 8003 cluster nodes

Springboot连接cluster:









*##redis配置***spring**:  
 *#redis配置* **redis**:  
 *# Redis数据库索引（默认为0）* **database**: 0  
 *#host: 127.0.0.1  
 #port: 6379* **password**: zhangjun  
 *# 集群环境打开下面注释，单机不需要打开* **cluster**:  
 *##集群信息* **nodes**: 192.168.234.128:8001,192.168.234.128:8002,192.168.234.128:8003,192.168.234.128:8004,192.168.234.128:8005,192.168.234.128:8006,192.168.234.128:8007,192.168.234.128:8008  
 *##默认值是5 一般当此值设置过大时，容易报：Too many Cluster redirections* **maxRedirects**: 3

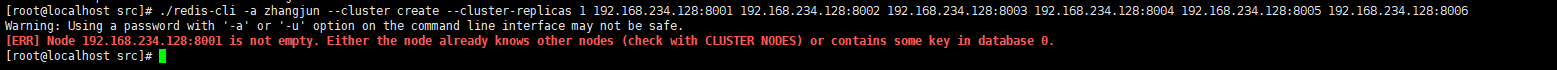
关闭集群后重启：

1. 删除集群8001-8008下的每个文件夹除开redis.conf文件都删除，重启实例，重启集群
2. 不删除任何文件，只是重启实例，redis会依靠nodes.conf自动构建原有集群，依靠aof、rdb等本地备份文件恢复数据库。

3、只删除nodes.conf文件，重新构建集群，启动实例，启动集群

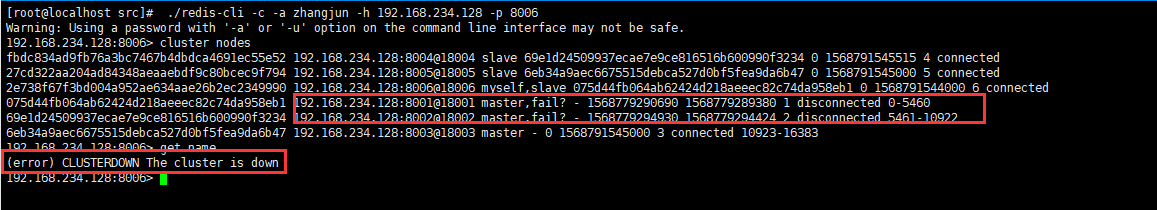
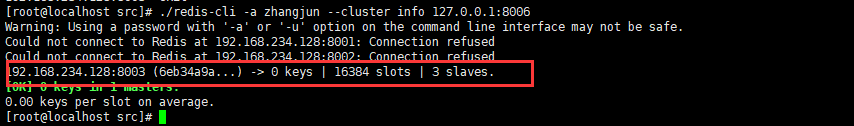
再次启动实例，启动集群报错，因为存在nodes.conf。不需要再次启动集群

Node 192.168.234.128:8001 is not empty. Either the node already knows other nodes (check with CLUSTER NODES) or contains some key in database 0



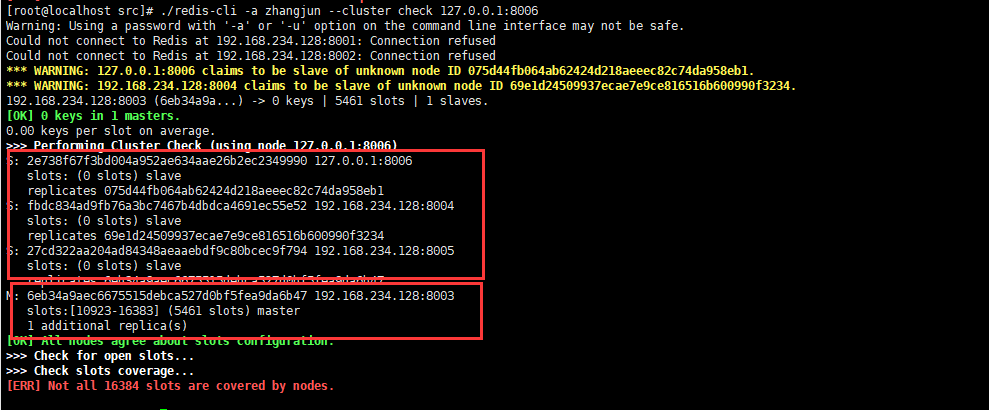
若：

CLUSTERDOWN The cluster is down解决办法：

检查集群状态slots详细分配。

[root@localhost src]# ./redis-cli -a zhangjun --cluster check 127.0.0.1:8006



提示：slot都被节点覆盖到，官方的建议是使用fix修复

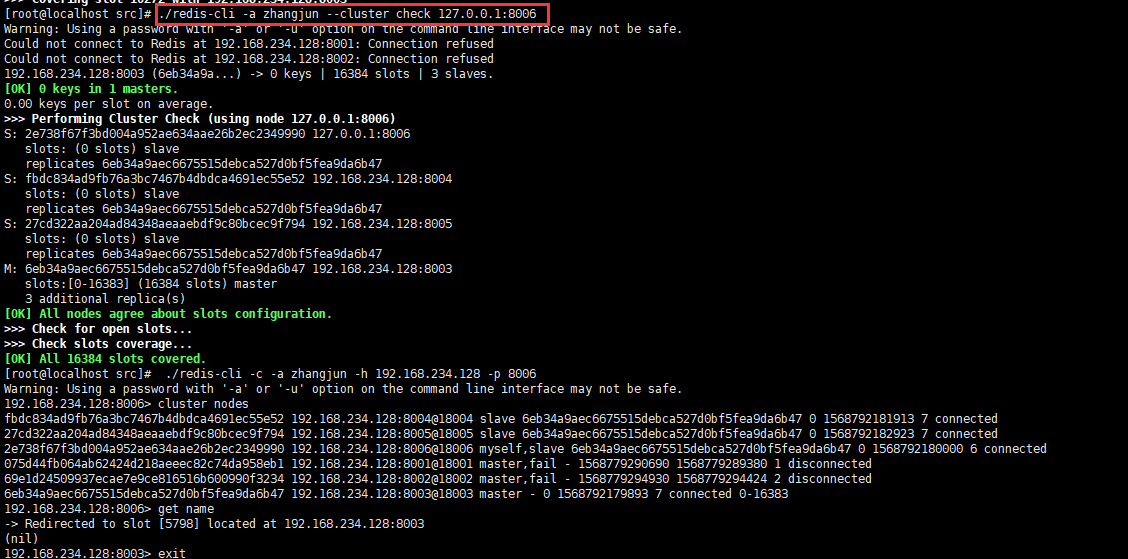
检查

[root@localhost src]# ./redis-cli -a zhangjun --cluster check 127.0.0.1:8006

修复：

[root@localhost src]# ./redis-cli -a zhangjun --cluster fix 127.0.0.1:8006

修复之后检查，集群正常：

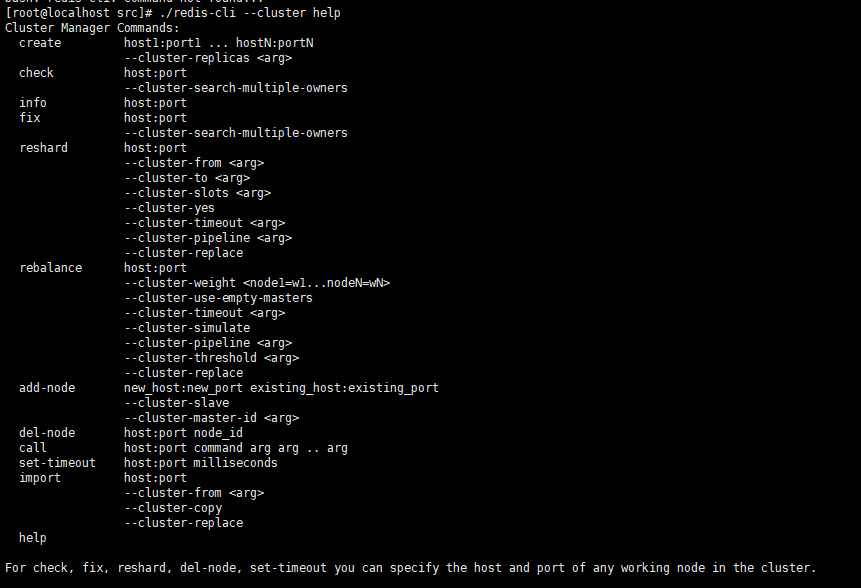


集群是多台机器做同一件事情(高性能HPC) 或者任意一台机器处理一个事情(负载均衡LBC)

主从属于高可用的范畴: 主机挂了 备用马上接替. 平时只使用主机(高可用HA)

查看redis-cli帮助：

[root@localhost src]# ./redis-cli --cluster help



Kill掉所有的redis进程：

[root@localhost 8006]# pkill -9 redis

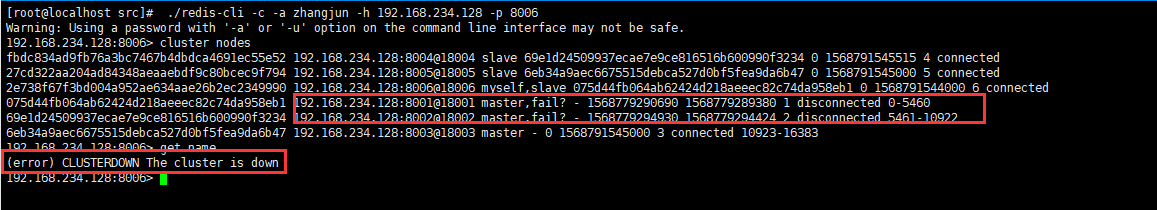
[root@localhost 8006]# ps -ef | grep redis



集群最少 三主三从

Redis的基础介绍与安装使用步骤：<https://www.jianshu.com/p/2a23257af57b>  
Redis的基础数据结构与使用：<https://www.jianshu.com/p/c95c8450c5b6>  
Redis核心原理：<https://www.jianshu.com/p/4e6b7809e10a>  
Redis 5 之后版本的高可用集群搭建：<https://www.jianshu.com/p/8045b92fafb2>  
Redis 5 版本的高可用集群的水平扩展：<https://www.jianshu.com/p/6355d0827aea>  
Redis 5 集群选举原理分析：<https://www.jianshu.com/p/e6894713a6d5>  
Redis 5 通信协议解析以及手写一个Jedis客户端：<https://www.jianshu.com/p/575544f68615>

猜测原因：集群数量不够，集群开启失败



批量替换：

:1,$s/8006/8008/g



批量删除：

[root@localhost 8002]# rm -rf appendonly.aof dump.rdb nodes-8002.conf redis.log

批量结束redis进程：

[root@localhost 8008]# pkill -9 redis





