# MongoDB集群配置

1. 准备：
   1. MongoDB v2.0.0
   2. 本次测试环境：

Server1:192.168.1.101(Win7 32bit)

Server2:192.168.1.105(Linux 64bit)

* 1. 创建数据目录
     1. Win7 ：mongoDB程序和data都放在C盘下
     2. Linux： mongoDB 放在/home/mongodb/mongodb-linux-x86\_64-2.0.0/ data放在根目录下

1. 配置
2. 配置分片shard1用到的replica sets(3 nodes)：

./mongod --shardsvr --replSet set1 --port 27017 --dbpath /data/shard11 --oplogSize 100 --logpath /data/shard11.log --logappend --fork

./mongod --shardsvr --replSet set1 --port 27018 --dbpath /data/shard21 --oplogSize 100 --logpath /data/shard21.log --logappend --fork

./mongod --shardsvr --replSet set1 --port 27019 --dbpath /data/shard31 --oplogSize 100 --logpath /data/shard31.log --logappend --fork

初始化replica set：

./mongo 192.168.1.101:27017

执行：

config={\_id:'set1',members:[{\_id:0,host:'192.168.1.101:27017'},{\_id:1,host:'192.168.1.101:27018'},{\_id:2,host:'192.168.1.101:27019'}]}

rs.initiate(config)

1. 配置分片shard2用到的replica sets(3 nodes)：

mongod.exe --shardsvr --replSet set2 --port 27017 --dbpath ../../data/shard12 -oplogSize 100 --logpath ../../data/shard12.log --logappend

mongod.exe --shardsvr --replSet set2 --port 27018 --dbpath ../../data/shard22 -oplogSize 100 --logpath ../../data/shard22.log --logappend

mongod.exe --shardsvr --replSet set2 --port 27019 --dbpath ../../data/shard32 -oplogSize 100 --logpath ../../data/shard32.log --logappend

初始化replica set：

mongo 192.168.1.105:27017

执行：

config={\_id:'set2',members:[{\_id:0,host:'192.168.1.105:27017'},{\_id:1,host:'192.168.1.105:27018'},{\_id:2,host:'192.168.1.105:27019'}]}

rs.initiate(config)

到此就配置好了二个replica sets，也就是准备好了二个shards。

1. 配置config server（一个配置服务器）

在server1上执行：

./mongod --configsvr --dbpath /data/config --port 20000 --logpath /data/config.log --fork

1. 配置mongos路由

在server1执行：

./mongos --configdb 192.168.1.101:20000 --port 30000 --chunkSize 2 --logpath /data/mongos.log --logappend --fork

在server2执行：

mongos --configdb 192.168.1.101:20000 --port 30000 --chunkSize 2 --logpath ../../data/mongos.log --logappend

1. 配置Shard Cluster

在server1执行：

./mongo 192.168.1.101:30000/admin

db.runCommand({addshard:"set1/192.168.1.101:27017,192.168.1.101:27018,192.168.1.101:27019",name:"s1",maxsize:20480,allowLocal:true } )

在server2执行：

mongo 192.168.1.105:30000/admin

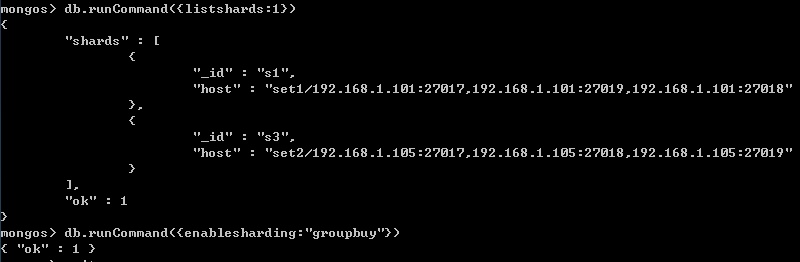
db.runCommand({addshard:"set2/192.168.1.105:27017,192.168.1.105:27018,192.168.1.105:27019",name:"s2",maxsize:20480,allowLocal:true } )

然后连接到任意一台mongos上执行：

./mongo 192.168.1.101:30000/admin

db.runCommand({listshards:1})

db.runCommand({enablesharding:"groupbuy"})



通过执行以上命令，可以让数据库跨shard，如果不执行这步，数据库只会存放在一个shard，一旦激活数据库分片，数据库中不同的collection将被存放在不同的shard上，但一个collection仍旧存放在同一个shard上，要使单个collection也分片，还需单独对collection作些操作。如下图，图1表示分片和复制集已经成功配置，图2表示sever1上27017端口运行的mongoDB为复制集set1 的primary 节点，图3同理。

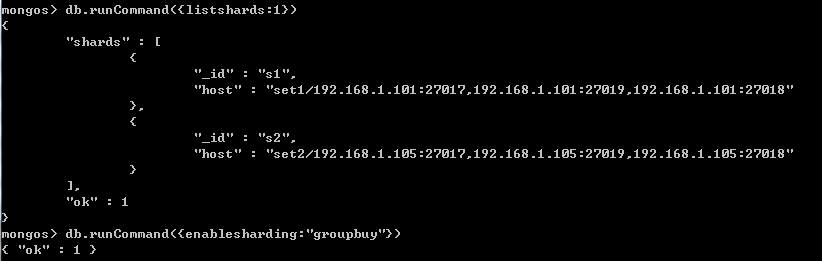


图1

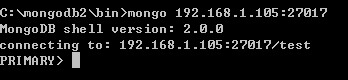


图2



图3

1. 测试

连接到任意一台mongos：

./mongo 192.168.1.101:30000/admin

use admin

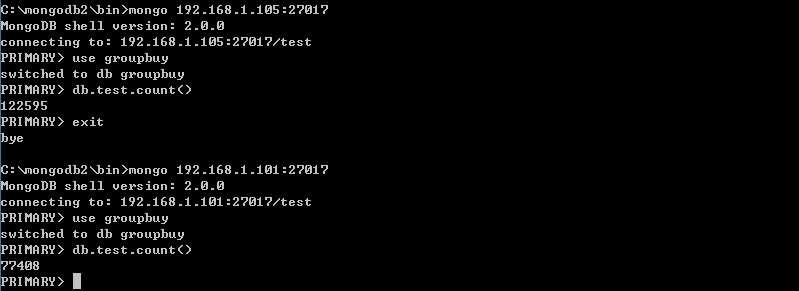
db.runCommand( { shardcollection : "groupbuy.test",key : {id: 1} } )

use groupbuy

for (var i = 1; i <= 200003; i++) db.test.save({id:i,value1:"1234567890",value2:"1234567890",value3:"1234567890"})

db.test.stats()

运行完后再分别进入两个分片mongoDB查看数据，可以看到数据确实分片存储了，数据之所以没有均匀分片，这是因为在配置mongos路由时参数--chunkSize 2，这里表示分片上的一块最小存储连续的数据块为2M。



在server1上执行 用kill -2 模拟server1上的27017端口上的primary节点down掉，结果复制集上的27019上的secondary节点自动变为了primary节点 ，当再启动27017mongoDB时，27017又自动变为secondary。可见，复制集生效。

