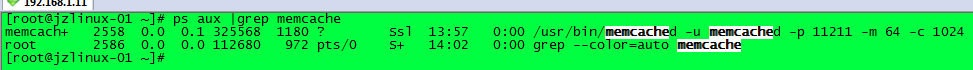
**一、安装memcached**

1、yum install -y memcached //安装包

2、systemctl start memcached //启动服务

3、查看服务是否起来

[root@jzlinux-01 ~]# ps aux |grep memcache



-u 用户名 ； -p 端口； -m 分配的内存大小，单位M； -c 最大的并发数。

4、vim /etc/sysconfig/memcached //编辑里面的内容可以更改相关参数。

PORT="11211"

USER="memcached"

MAXCONN="1024"

CACHESIZE="64"

OPTIONS="" //监听的主机

**二、查看memcached状态**

1、memcached-tool 127.0.0.1:11211 stats //查看状态

2、yum install -y nc

echo stats |nc 127.0.0.1 11211 //查看状态

3、yum install -y libmemcached

memstat --servers=127.0.0.1:11211 //查看状态

cmd\_get表示总的get次数

get\_hits表示get的总命中次数

命中率 = get\_hits/cmd\_get

**三、memcached命令行**

1、telnet 127.0.0.1 11211

set key2 0 30 2 //key2是键的名称；0是一个16位的无符号的整数(以十进制的方式表示)。

30是过期时间秒；2是存储数据需为2个字节.

2、存储

ab

STORED

get key2

VALUE key2 0 2

ab

END



3、 <command name> <key> <flags> <exptime> <bytes>\r\n <data block>\r\n

set|add|replace key2 0 30 2 回车 ab 回车

set表示按照相应的<key>存储该数据，没有的时候增加，有的时候覆盖

add表示按照相应的<key>添加该数据,但是如果该<key>已经存在则会操作失败

replace表示按照相应的<key>替换数据,但是如果该<key>不存在则操作失败。

4、delete key2 //删除值

**四、memcached数据导出和导入**

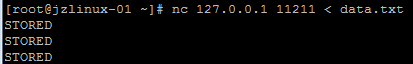
ctrl + ] //退出telnet

1、memcached-tool 127.0.0.1:11211 dump > data.txt //数据导出

2、重启设服务后导入数据

systemctl restart memcached

nc 127.0.0.1 11211 < data.txt



cat data.txt

add name 1 1520308320 3

yjz

add age 1 1520308320 2

30

add sex 1 1520308320 3

man

1520308320:时间戳，超过这个时间戳，数据无法导入。

date -d @1520308320 //时间戳对应的具体时间



为了避免导入时，时间戳过期，需手动更改时间戳，再导入

date -d "+1 hour" +%s // 在当前时间下，把时间后延一个小时。



**五、php连接memcached**

wget http://www.apelearn.com/bbs/data/attachment/forum/memcache-2.2.3.tgz

1、tar zxf memcache-2.2.3.tgz

2、cd memcache-2.2.3

3、/usr/local/php-fpm/bin/phpize

4、./configure --with-php-config=/usr/local/php-fpm/bin/php-config

5、make && make install

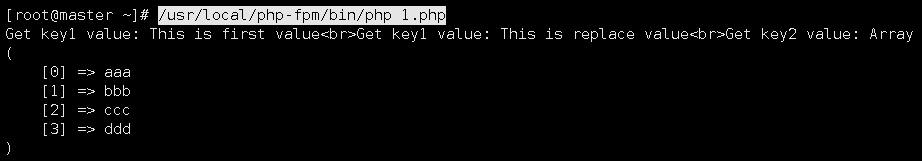
6、vim /usr/local/php-fpm/etc/php.ini //添加下面的内容

extension=memcache.so

7、/usr/local/php-fpm/bin/php -m|grep memcache

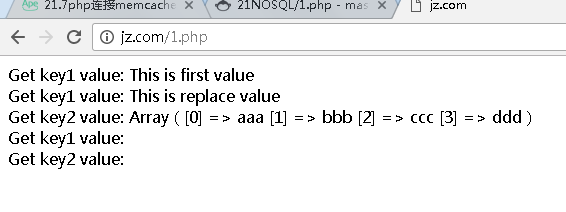


8、/usr/local/php-fpm/bin/php 1.php //测试



或者

/data/wwwroot/jz.com //将1.php拷贝到相应网站的根目录（注意设置win的hosts）



**Memcached中存储session**

1、在php-fpm pool中设置

php\_value[session.save\_handler] = memcache

php\_value[session.save\_path] = " tcp://192.168.1.18:11211 "

2、cd /usr/local/php-fpm/etc/php-fpm.d //pool的位置，一个网站一个pool

[root@master php-fpm.d]# ls

jz.conf www.conf

[root@master php-fpm.d]# cat jz.conf

[JZ]

listen = /tmp/JZ.sock //自定义的socket

#listen = 127.0.0.1:9000

listen.mode = 666

user = php-fpm

group = php-fpm

pm = dynamic

pm.max\_children = 50

pm.start\_servers = 20

pm.min\_spare\_servers = 5

pm.max\_spare\_servers = 35

pm.max\_requests = 500

rlimit\_files = 1024

php\_value[session.save\_handler] = memcache

php\_value[session.save\_path] = "tcp://192.168.1.18:11211"

[root@master php-fpm.d]# cat www.conf

[www]

listen = /tmp/php-fcgi.sock //本地socket

#listen = 127.0.0.1：9000

listen.mode = 666

user = php-fpm

group = php-fpm

pm = dynamic

pm.max\_children = 50

pm.start\_servers = 20

pm.min\_spare\_servers = 5

pm.max\_spare\_servers = 35

pm.max\_requests = 500

rlimit\_files = 1024

php\_value[session.save\_handler] = memcache

php\_value[session.save\_path] = "tcp://127.0.0.1:11211"

3、/usr/local/nginx/conf/vhost //虚拟主机配置文件

cat jz.com.conf

server

{

listen 80;

server\_name jz.com;

index index.html index.htm index.php;

root /data/wwwroot/jz.com;

location ~ \.php$

{

include fastcgi\_params;

fastcgi\_pass unix:/tmp/JZ.sock;

fastcgi\_index index.php;

fastcgi\_param SCRIPT\_FILENAME /data/wwwroot/jz.com$fastcgi\_script\_name;

}

}

[root@master vhost]# cat test.com.conf

server

{

listen 80;

server\_name test.com;

index index.html index.htm index.php;

root /data/wwwroot/test.com;

location ~ \.php$

{

include fastcgi\_params;

fastcgi\_pass unix:/tmp/php-fcgi.sock;

fastcgi\_index index.php;

fastcgi\_param SCRIPT\_FILENAME /data/wwwroot/test.com$fastcgi\_script\_name;

}

}

4、vim /etc/hosts //非本地池子的socket要在这里定义

192.168.1.18 jz.com

5、cd /data/wwwroot //网站跟目录

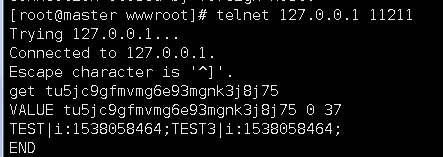
curl jz.com/session.php



curl test.com/session.php







**redis安装**

1、去redis.io,下载

cd /usr/local/src

wget http://download.redis.io/releases/redis-4.0.9.tar.gz

2、tar zxvf redis-4.0.9.tar.gz //解压

make && make install //安装过程中如果报错

yum install gcc

make MALLOC=libc

3、cp redis.conf /etc/redis.conf //拷贝配置文件

vim /etc/redis.conf //将"no"（在前台启动） 改成 "yes"(在后台启动)

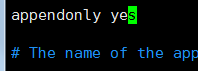


定义日志路径



dump.rdb 存放的路径和appendonly.aof存放路径





4、mkdir -p /data/redis //创建目录

sysctl vm.overcommit\_memory=1

echo never > /sys/kernel/mm/transparent\_hugepage/enabled

5、redis-server /etc/redis.conf //开启服务

**redis持久化**

1、vim /etc/redis.conf //满足下面任一条件，就将数据写入磁盘

#save ""

save 900 1 //900秒发生一次改变

save 300 10

save 60 10000

2、关闭持久化

save ""

#save 900 1

#save 300 10

#save 60 10000

**redis数据类型**

1、sting类型

redis-cli //进入redis

set mykey 123 //设置key的名字为mykey，value的值为123，进入redis后的操作

get mykey //根据key查找value

mset k1 1 k2 2 k3 a //批量设置key和对应的value

mget k1 k2 k3 //批量显示key的值

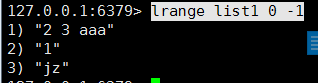
2、list类型 链表结构

lpush list1 "jz" //list1是名字，jz是它的值

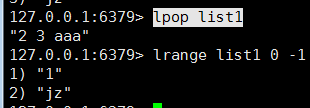
lpush list1 "1"

lpush list1 "2 3 aaa" //都是给他赋值

lrange list1 0 -1 //取出list1的所有值，最先赋的值，取出的时候放到了最后



lpop list1 //取出数据，取完数据后，该数据就不在了



3、set类型，集合

设置一个set类型的数据，名字为set1,后面跟的是它的值

sadd set1 a

sadd set1 b

sadd set1 c

设置一个set类型的数据，名字为set2,后面跟的是它的值

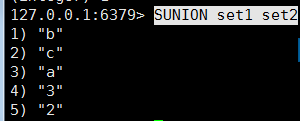
SADD set2 a

SADD set2 2

SADD set2 3

SADD set2 c

SUNION set1 set2 //求并集（就是2者全部的值，不考虑重复内容）



SINTER set1 set2 //求交集（二者相同的值）



SDIFF set1 set2 //差集，就是set1里面有的值，在set2里面没有



就是set2里面有的值，在set1里面没有



SREM set1 c //删除set1的元素

sort set //有序集合的类型

设置集合的名字set3，以及后面的元素

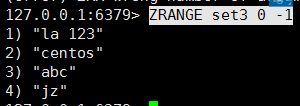
ZADD set3 12 abc

ZADD set3 2 "la 123"

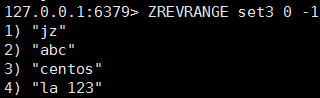
ZADD set3 24 "jz"

ZADD set3 4 "centos"

ZRANGE set3 0 -1 //按照数字从小到大排序



ZREVRANGE set3 0 -1 //按照数字从大到小排序



hash类型

HSET hash1 name jz

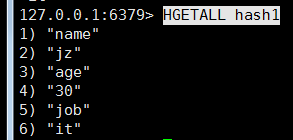
HSET hash1 age 30

HSET hash1 job it

HGET hash1 job //显示相关的元素



HGETALL hash1 //显示所有元素



**redis常用操作()**

1、set key1 jz

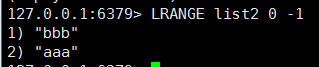
set key1 JZ //第二次赋值会覆盖第一次的

2、SETNX key1 fff //检测key1是否已经存在，fff自定义的，并不是key1的值，如果存在返回0，不存在返回1，并且创建key1,fff就是新创建key1的值

3、setex key3 10 jz //key3的值为jz，过期时间10s

4、lpush list2 aaa //在左侧加入一个元素

lpush list2 bbb //后插入的排在前面



5、LPOP list2 //取最上面的数据，取完后数据没有了

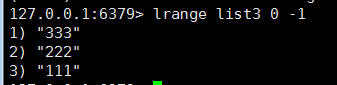


6、在左侧加数据，及在最上面加

lpush list3 111

lpush list3 222

lpush list3 333



rpop list3 //从下面开始取数据，取完后这个数据就没有了

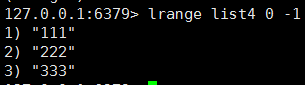


7、在右侧加入数据，就是在下面加数据

rpush list4 111

rpush list4 222

rpush list4 333



8、LINSERT list4 after 111 2 //在list4的111（元素的值）后面插入一个2，如果有相同的元素值，会选择第一个元素后面插入

LINSERT list4 before 111 2 //在111前面插入一个2

9、lset list5 5 6 //将list5的第5（从0开始数）个元素的值改为6

10、LINDEX list5 5 //查看第5（从0开始数）个元素的值

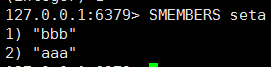
11、LLEN list5 //查看list5中有多少个元素

8、

sadd seta aaa

sadd seta bbb

SMEMBERS seta



srem seta aaa //删除元素‘aaa’

spop seta //随机取出一个元素，并且删除

sdiff seta setb //求2者的差集

SDIFFSTORE setc seta setb //将seta和setb的差集存入setc

sinter seta setb //求交集

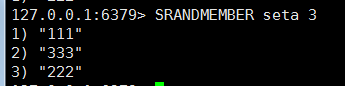
sinterstore setd seta setb //交集存入setd

SUNION seta setb //求并集

sunionstore sete seta setb //求并集，并且存入sete

SISMEMBER seta 111 //判断‘111’元素是否在seta中，在返回‘1’，不在返回‘0’

SRANDMEMBER seta 3 //随机取出seta的3个元素，不删除，不加‘3’默认是取一个



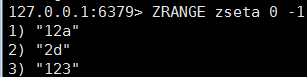
zadd //创建有序集合，[score namer]，根据score来排序

ZRANGE zseta 0 -1 //查看所有元素，默认按照score的由小到大排序

zadd zseta 11 123

zadd zseta 0 12a

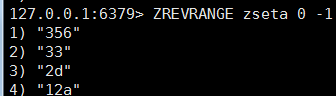
zadd zseta 5 2d



ZREM zseta 123 //删除指定元素，‘123’为元素值

ZRANK zseta 12a //返回元素索引值（从上到下，且是正向排序）

ZREVRANGE zseta 0 -1 //反向排序（score排在最上面）



ZREVRANK zseta 356 //反向排序（score最大的在上面）元素的下标值

ZCARD zseta //查看集合中有几个元素

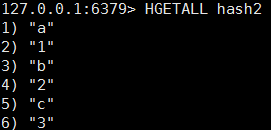
ZCOUNT zseta 10 30 //在10-30这个分值区间内有多少个元素

ZRANGEBYSCORE zseta 10 30 //查看socre在10-30之间有哪些元素

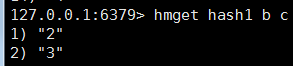
ZREMRANGEBYRANK zsetb 0 1 //删除下标为0到1的元素值（正向排序）

hmset hash2 a(field) 1(value) b 2 c 3 //[field value]

hgetall hash2



hmget hash2 b c //根据filed找出对应的value



hdel hash2 a //删除指定的filed及对应的value

hkeys hash2 //打印所有的key

HVALS hash2 //打印所有的value

hlen hash2 //查看hash2有多少个filed

**redis操作键值**

keys \* //取出所有的key

keys my\* //模糊匹配

EXISTS key1 //查看key1这个键值是否存在，存在返回1，不存在返回0

del key1 //删除key1

EXPIRE key2 10 //给key2设置过期时间

TTL set2 //查看set2的过期时间

select 1 //切换数据库，默认是0库

move zsetb 1 //将zsetb 从当前库移到1库

PERSIST set3 //取消set3的过期时间

RANDOMKEY //随机返回一个key

RENAME seta sett //将seta重命名为sett

type sett //查看sett的类型

dbsize//返回当前数据库中key的数目



FLUSHdb //清空当前数据库中的所有key

flushall //清空所有数据库中的所有key

bgsave //保存数据到 rdb文件中，在后台运行

CONFIG set timeout 100 //更改timount的时间为100

关于redis的rdb,恢复

a、在redis中保存数据，bgsave

b、数据保存于config get dir（会出现目录）dump.rdb中

c、将该目录的dump.rdb文件备份到其他地方

d、关闭aof，默认是走aof恢复的



e、恢复时，将备份的dump.rdb文件拷贝到config get dir（会出现目录），重启服务即可

**redis安全设置**

1、vim /etc/redis.conf //设置密码

requirepass jz@3378 //后面的是密码

2、redis-cli -a 'jz@3378' //设置密码后登录

rename-command CONFIG jz //在配置文件里面将CONFIG改成jz

rename-command CONFIG "" //如果改成这样就禁止掉了config这个命令

**redis慢查询日志**

slowlog-log-slower-than 1000 //单位ms，表示慢于1000ms则记录日志

slowlog-max-len 128 //定义日志长度，表示最多存128条

slowlog get //列出所有的慢查询日志

slowlog get 2 //只列出2条

slowlog len //查看慢查询日志条数

**redis存储session**

1、vim /usr/local/php-fpm/etc/php.ini//更改或增加

session.save\_handler = "redis"

session.save\_path = "tcp://127.0.0.1:6379"

2、或者apache虚拟主机配置文件中也可以这样配置：

php\_value session.save\_handler " redis"   
php\_value session.save\_path " tcp://127.0.0.1:6379"

3、或者php-fpm配置文件对应的pool中增加：

php\_value[session.save\_handler] = redis

php\_value[session.save\_path] = " tcp://127.0.0.1:6379 "

测试：

1、在vim /usr/local/apache2.4/conf/extra/httpd-vhosts.conf //添加

php\_value session.save\_handler "redis"   
php\_value session.save\_path "tcp://127.0.0.1:6379"

2、重启apache服务

3、在 /data/wwwroot/abc.com （crul localhost，默认的地方）

添加1.php

<?php

session\_start();

if (!isset($\_SESSION['TEST'])) {

$\_SESSION['TEST'] = time();

}

$\_SESSION['TEST3'] = time();

print $\_SESSION['TEST'];

print "<br><br>";

print $\_SESSION['TEST3'];

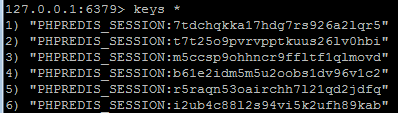
print "<br><br>";

print session\_id();

?>

在开启redis的情况下，crul localhost/1.php

正的话进入redis后 keys \* 会出现数据



**redis主从配置**

1、主上配置

bind 127.0.0.1 注释掉

daemonize yes //默认是no

protected-mode no //默认是yes

logfile "/var/log/redis.log"

dir /data/redis //设置好后，去服务器上创建这个目录

requirepass jz@3378 //后面的是密码

2、从上配置

bind 127.0.0.1 注释掉

daemonize yes //默认是no

protected-mode no //默认是yes

logfile "/var/log/redis.log"

dir /data/redis //设置好后，去服务器上创建这个目录

slaveof 192.168.1.37 6379

masterauth jz@3378 //认证主的密码

完成后，启动服务，主上所有的数据，自动同步到从上去，且默认从不可写

**redis集群搭建配置**

1、cd /etc //在第一台服务器上

vim redis\_7000.conf

port 7000

bind 192.168.1.37

daemonize yes

pidfile /var/run/redis\_7000.pid

dir /data/redis\_data/7000

cluster-enabled yes

cluster-config-file nodes\_7000.conf

cluster-node-timeout 10100

appendonly yes

############################

vim redis\_7002.conf

port 7002

bind 192.168.1.37

daemonize yes

pidfile /var/run/redis\_7002.pid

dir /data/redis\_data/7002

cluster-enabled yes

cluster-config-file nodes\_7002.conf

cluster-node-timeout 10100

appendonly yes

##################################

vim redis\_7004.conf

port 7004

bind 192.168.1.37

daemonize yes

pidfile /var/run/redis\_7004.pid

dir /data/redis\_data/7004

cluster-enabled yes

cluster-config-file nodes\_7004.conf

cluster-node-timeout 10100

appendonly yes

#####################

mkdir /data/redis\_data

mkdir /data/redis\_data/{7000,7002,7004}

2、在第二台服务器上配置 cd /etc

vim redis\_7001.conf

port 7001

bind 192.168.1.17

daemonize yes

pidfile /var/run/redis\_7001.pid

dir /data/redis\_data/7001

cluster-enabled yes

cluster-config-file nodes\_7001.conf

cluster-node-timeout 10100

appendonly yes

####################

vim redis\_7003.conf

port 7003

bind 192.168.1.17

daemonize yes

pidfile /var/run/redis\_7003.pid

dir /data/redis\_data/7003

cluster-enabled yes

cluster-config-file nodes\_7003.conf

cluster-node-timeout 10100

appendonly yes

######################

vim redis\_7005.conf

port 7005

bind 192.168.1.17

daemonize yes

pidfile /var/run/redis\_7005.pid

dir /data/redis\_data/7005

cluster-enabled yes

cluster-config-file nodes\_7005.conf

cluster-node-timeout 10100

appendonly yes

##########################

mkdir /data/redis\_data

mkdir /data/redis\_data/{7001,7003,7005}

3、启动服务

redis-server /etc/redis\_7000.conf

redis-server /etc/redis\_7002.conf

redis-server /etc/redis\_7004.conf

4、启动服务

redis-server /etc/redis\_7001.conf

redis-server /etc/redis\_7003.conf

redis-server /etc/redis\_7005.conf

5、安装ruby2.2 （只需要一台机器上运行），我安装在第一台服务器上

a、yum -y groupinstall "Development Tools"

b、yum -y install gdbm-devel libdb4-devel libffi-devel libyaml libyaml-devel ncurses-devel openssl-devel readline-devel tcl-deve

cd /root

mkdir -p rpmbuild/{BUILD,BUILDROOT,RPMS,SOURCES,SPECS,SRPMS}

c、wget http://cache.ruby-lang.org/pub/ruby/2.2/ruby-2.2.3.tar.gz -P rpmbuild/SOURCES

wget https://raw.githubusercontent.com/tjinjin/automate-ruby-rpm/master/ruby22x.spec -P rpmbuild/SPECS

d、rpmbuild -bb rpmbuild/SPECS/ruby22x.spec

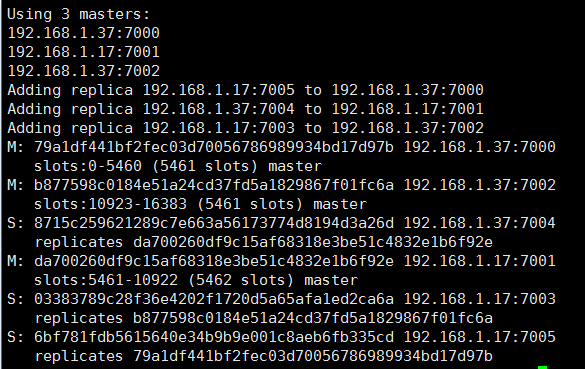
e、yum -y localinstall rpmbuild/RPMS/x86\_64/ruby-2.2.3-1.el7.centos.x86\_64.rpm

f、gem install redis

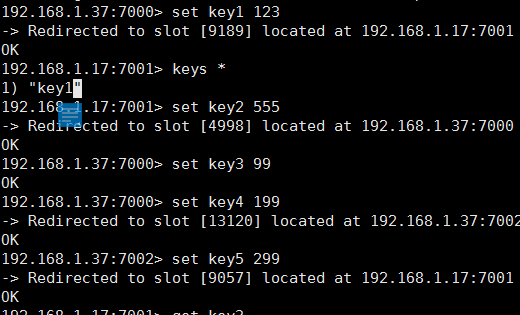
g、cp /usr/local/src/redis-4.0.9/src/redis-trib.rb /usr/bin

h、redis-trib.rb create --replicas 1 192.168.1.37:7000 192.168.1.37:7002 192.168.1.37:7004 192.168.1.17:7001 192.168.1.17:7003 192.168.1.17:7005

默认，前3个端口会被设置成master



redis-cli -c -h 192.168.1.37 -p 7000 //连接到这台上去



A、创建的这些键值不是存在同一台设备上的，是分布式的

B、redis-trib.rb check 192.168.1.37:7000 //查看状态

C、redis-cli -c -h 192.168.1.37 -p 7000 //连接到这台上去

D、CLUSTER NODES //查看节点信息

E、CLUSTER INFO //查看集群信息

:1,$s/i/I/g //vim 里面的替换所有的i成I

F、CLUSTER MEET 192.168.1.37 7006 //添加节点 IP PORT

G、将当前节点设置为指定节点的slave

redis-cli -c -h 192.168.1.37 -p 7006 //登录到当前节点

CLUSTER REPLICATE 76310af79f6bb4bb1b7358751fdd9bf6cb21e2c8 //后面是指定节点的node\_id，可以通过CLUSTER NODES查看到

8e7c7bbf316f86c01ac3ab11dd2bea153684d509 192.168.1.37:7006@17006 myself,slave 76310af79f6bb4bb1b7358751fdd9bf6cb21e2c8 //这个可以看出7006变成了slave,他的master的node\_id为76310af79f6bb4bb1b7358751fdd9bf6cb21e2c8

H、cluster forget node\_id //移除节点，不能移除自己和master,要移除master需先将他变成slave

CLUSTER FORGET 8e7c7bbf316f86c01ac3ab11dd2bea153684d509

CLUSTER SAVECONFIG //保存配置

**mongodb**

**一、安装**

https://docs.mongodb.com/manual/tutorial/install-mongodb-on-red-hat/ //官方安装文档

1、cd /etc/yum.repos.d/

2、vim mongodb-org-3.6.repo

[mongodb-org-3.6]

name=MongoDB Repository

baseurl=https://repo.mongodb.org/yum/redhat/$releasever/mongodb-org/3.6/x86\_64/

gpgcheck=1

enabled=1

gpgkey=https://www.mongodb.org/static/pgp/server-3.6.asc

3、yum install -y mongodb-org //安装

**二、连接mongodb**

vim /etc/mongod.conf //绑定多个IP用逗号分隔。



1、systemctl start mongod //启动服务

2、mongo --host 192.168.1.106 --port 27017 //连接服务

3、mongo -uusername -ppasswd --authenticationDatabase db //验证时登录

**三、 mongodb用户管理**

1、use admin //切换数据库

db.createUser( { user: "admin", customData: {description: "superuser"}, pwd: "admin122", roles: [ { role: "root", db: "admin" } ] } )

user指定用户，customData为说明字段，可以省略，pwd为密码，roles指定用户的角色，db指定库名

2、db.system.users.find() //切换到admin库后，可以列出所有的用户

3、show users //查看当前库下的用户

4、db.dropUser('aming') //删除用户

5、vim /usr/lib/systemd/system/mongod.service //加入--auth,再次登录服务需要用户认证了，要输入用户和密码了



systemctl daemon-reload

systemctl restart mongod

6、mongo --host 192.168.1.106 --port 27017 -uadmin -p 'admin122' --authenticationDatabase "admin"

7、use db1

db.createUser( { user: "test1", pwd: "123aaa", roles: [ { role: "readWrite", db: "db1" }, {role: "read", db: "db2" } ] } )

8、use db1

switched to db db1

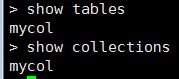
> db.auth('test1','123aaa') //验证普通用户

9、在这个普通用户里面创建集合

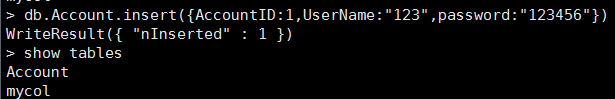
db.createCollection("mycol", { capped : true, size : 6142800, max : 10000 } )



10、show tables或者show collections //查看集合

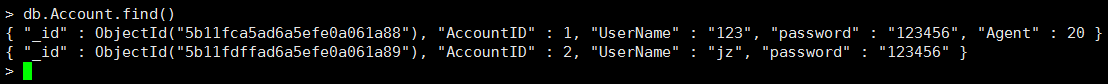


11、db.Account.insert({AccountID:1,UserName:"123",password:"123456"}) //如果集合不存在，直接插入数据，则mongodb会自动创建集合



12、db.Account.update({AccountID:1},{"$set":{"Agent":20}}) //更新数据

13、db.Account.find() //查看数据



14、db.Account.find({AccountID:2}) //查看指定条件的



15、db.Account.remove({AccountID:1}) //删除指定数据

16、db.Account.drop() //删除所有文档，即删除集合

17、db.printCollectionStats() // 然后查看集合状态

**PHP的mongodb扩展**

1、mongodb扩展

cd /usr/local/src/

wget https://pecl.php.net/get/mongodb-1.3.0.tgz

tar zxvf mongodb-1.3.0.tgz

cd mongodb-1.3.0

/usr/local/php/bin/phpize //找到phpize这个命令

./configure --with-php-config=/usr/local/php/bin/php-config

make && make install

vi /usr/local/php/etc/php.ini //增加 extension = mongodb.so

/usr/local/php/bin/php -m



2、mongod扩展

cd /usr/local/src/

wget https://pecl.php.net/get/mongo-1.6.16.tgz

tar zxvf mongo-1.6.16.tgz

cd /usr/local/src/mongo-1.6.16

/usr/local/php/bin/phpize

./configure --with-php-config=/usr/local/php/bin/php-config

make && make install

vi /usr/local/php/etc/php.ini //增加 extension = mongo.so

/usr/local/php/bin/php -m



在 /data/wwwroot/abc.com （crul localhost，默认的地方） //测试

添加1.php

集合创建成功[root@bogon abc.com]# cat 1.php

<?php

$m = new MongoClient();

$db = $m->test;

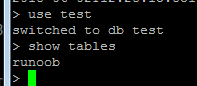
$collection = $db->createCollection("runoob");

echo "集合创建成功";

?>



登录mongodb确认，先关闭用户认证



**mongodb副本集**

1、三台设备，一主二从，三台设备均需要安装mongodb(详见安装)

主：192.168.1.113

从：192.168.1.107

从:192.168.1.111

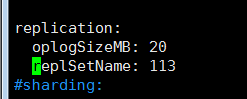
2、vim /etc/mongod.conf

bindIp: 127.0.0.1,192.168.1.113 //监听的IP

replication:

oplogSizeMB: 20

replSetName: 113 //113是自定义的



3、在107从上操作

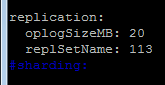
vim /etc/mongod.conf

bindIp: 127.0.0.1,192.168.1.107 //监听的IP

replication:

oplogSizeMB: 20

replSetName: 113 //113是自定义的



4、在111从上操作

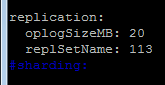
vim /etc/mongod.conf

bindIp: 127.0.0.1,192.168.1.111 //监听的IP

replication:

oplogSizeMB: 20

replSetName: 113 //113是自定义的



5、在主上操作

mongo //在主上登录

>use admin

> config={\_id:"113",members:[{\_id:0,host:"192.168.1.113:27017"},{\_id:1,host:"192.168.1.107:27017"},{\_id:2,host:"192.168.1.111:27017"}]}

6、rs.initiate(config) //初始化数据

7、rs.status() //查看状态

113:PRIMARY> rs.status()

{

"set" : "113",

"date" : ISODate("2018-06-03T03:20:14.921Z"),

"myState" : 1,

"term" : NumberLong(1),

"heartbeatIntervalMillis" : NumberLong(2000),

"optimes" : {

"lastCommittedOpTime" : {

"ts" : Timestamp(1527996007, 1),

"t" : NumberLong(1)

},

"readConcernMajorityOpTime" : {

"ts" : Timestamp(1527996007, 1),

"t" : NumberLong(1)

},

"appliedOpTime" : {

"ts" : Timestamp(1527996007, 1),

"t" : NumberLong(1)

},

"durableOpTime" : {

"ts" : Timestamp(1527996007, 1),

"t" : NumberLong(1)

}

},

"members" : [

{

"\_id" : 0,

"name" : "192.168.1.113:27017",

"health" : 1,

"state" : 1,

"stateStr" : "PRIMARY",

"uptime" : 1710,

"optime" : {

"ts" : Timestamp(1527996007, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T03:20:07Z"),

"electionTime" : Timestamp(1527995293, 1),

"electionDate" : ISODate("2018-06-03T03:08:13Z"),

"configVersion" : 2,

"self" : true

},

{

"\_id" : 1,

"name" : "192.168.1.107:27017",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 732,

"optime" : {

"ts" : Timestamp(1527996007, 1),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1527996007, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T03:20:07Z"),

"optimeDurableDate" : ISODate("2018-06-03T03:20:07Z"),

"lastHeartbeat" : ISODate("2018-06-03T03:20:13.462Z"),

"lastHeartbeatRecv" : ISODate("2018-06-03T03:20:14.761Z"),

"pingMs" : NumberLong(2),

"configVersion" : 2

},

{

"\_id" : 2,

"name" : "192.168.1.111:27017",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 732,

"optime" : {

"ts" : Timestamp(1527996007, 1),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1527996007, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T03:20:07Z"),

"optimeDurableDate" : ISODate("2018-06-03T03:20:07Z"),

"lastHeartbeat" : ISODate("2018-06-03T03:20:13.631Z"),

"lastHeartbeatRecv" : ISODate("2018-06-03T03:20:14.762Z"),

"pingMs" : NumberLong(19),

"configVersion" : 2

}

],

"ok" : 1,

"operationTime" : Timestamp(1527996007, 1),

"$clusterTime" : {

"clusterTime" : Timestamp(1527996007, 1),

"signature" : {

"hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),

"keyId" : NumberLong(0)

}

}

}

**测试**

1、主上建库建集合测试

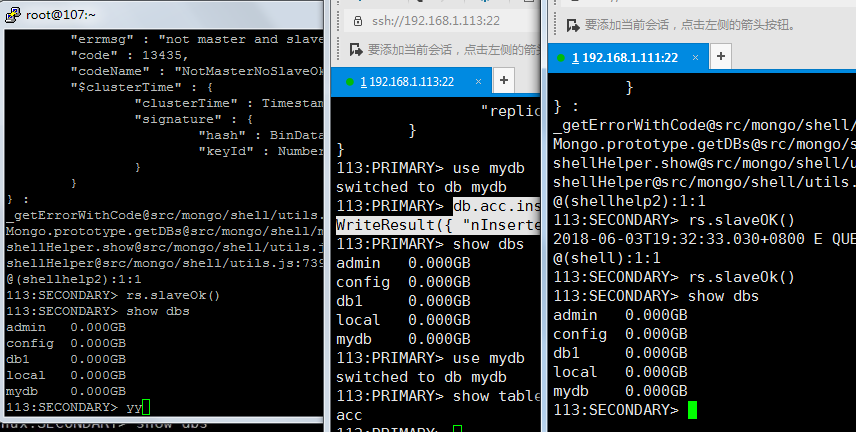
use mydb

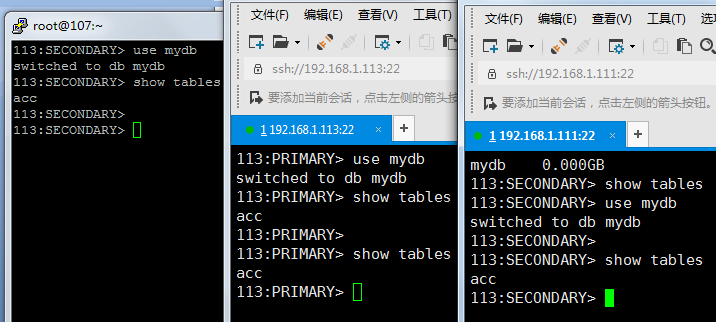
db.acc.insert({AccountID:1,UserName:"123",password:"123456"})

WriteResult({ "nInserted" : 1 })

2、在主上创建完后，登录到从上测试，数据都同步了

rs.slaveOk() //用这个命令解决，2个从上都要用这个命令解决





3、测试修改权重，来选举新的主，默认他们的权重都是1，设置新权重后，权重数字大的将选举成新的主

rs.config() //这个命令可以看到设备的权重



PRIMARY> cfg = rs.conf()

113:PRIMARY> cfg.members[0].priority = 3

3

113:PRIMARY> cfg.members[1].priority = 1 //改变权重后，它会变成主

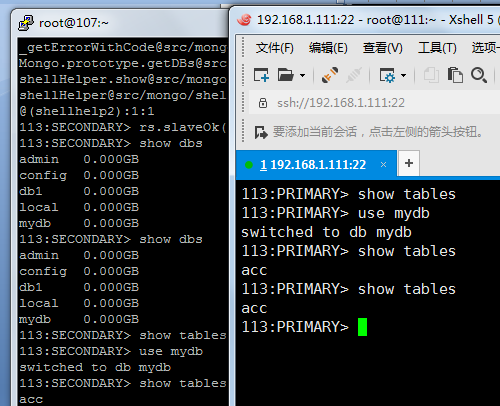
1

113:PRIMARY> cfg.members[2].priority = 2

2

rs.reconfig(cfg)

4、iptables -I INPUT -p tcp --dport 27017 -j DROP //主上禁止掉27017端口，111选举成新的主



**mongodb分片**

三台机器 A B C

A搭建：mongos、config server、副本集1主节点、副本集2仲裁、副本集3从节点

B搭建：mongos、config server、副本集1从节点、副本集2主节点、副本集3仲裁

C搭建：mongos、config server、副本集1仲裁、副本集2从节点、副本集3主节点

端口分配：mongos 20000、config 21000、副本集1 27001、副本集2 27002、副本集3 27003

1、分别在三台机器上创建各个角色所需要的目录

mkdir -p /data/mongodb/mongos/log

mkdir -p /data/mongodb/config/{data,log}

mkdir -p /data/mongodb/shard1/{data,log}

mkdir -p /data/mongodb/shard2/{data,log}

mkdir -p /data/mongodb/shard3/{data,log}

2、mkdir /etc/mongod/ //三台设备上均要操作

3、vim /etc/mongod/config.conf //三台设备均要同样操作

pidfilepath = /var/run/mongodb/configsrv.pid

dbpath = /data/mongodb/config/data

logpath = /data/mongodb/config/log/congigsrv.log

logappend = true

bind\_ip = 192.168.1.113 //这个ip根据设备IP改动

port = 21000

fork = true

configsvr = true #declare this is a config db of a cluster;

replSet=configs #副本集名称

maxConns=20000 #设置最大连接数

4、启动三台机器的config server

mongod -f /etc/mongod/config.conf

5、登录任意一台机器的21000端口，初始化副本集

mongo --host 192.168.1.113 --port 21000

use admin

config = { \_id: "configs", members: [ {\_id : 0, host : "192.168.1.113:21000"},{\_id : 1, host : "192.168.1.107:21000"},{\_id : 2, host : "192.168.1.111:21000"}]}

6、rs.initiate(config)

rs.status()

{

"set" : "configs",

"date" : ISODate("2018-06-03T05:36:38.714Z"),

"myState" : 1,

"term" : NumberLong(1),

"configsvr" : true,

"heartbeatIntervalMillis" : NumberLong(2000),

"optimes" : {

"lastCommittedOpTime" : {

"ts" : Timestamp(1528004189, 2),

"t" : NumberLong(1)

},

"readConcernMajorityOpTime" : {

"ts" : Timestamp(1528004189, 2),

"t" : NumberLong(1)

},

"appliedOpTime" : {

"ts" : Timestamp(1528004189, 2),

"t" : NumberLong(1)

},

"durableOpTime" : {

"ts" : Timestamp(1528004189, 2),

"t" : NumberLong(1)

}

},

"members" : [

{

"\_id" : 0,

"name" : "192.168.1.113:21000",

"health" : 1,

"state" : 1,

"stateStr" : "PRIMARY",

"uptime" : 1001,

"optime" : {

"ts" : Timestamp(1528004189, 2),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T05:36:29Z"),

"electionTime" : Timestamp(1528003967, 1),

"electionDate" : ISODate("2018-06-03T05:32:47Z"),

"configVersion" : 1,

"self" : true

},

{

"\_id" : 1,

"name" : "192.168.1.107:21000",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 242,

"optime" : {

"ts" : Timestamp(1528004189, 2),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1528004189, 2),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T05:36:29Z"),

"optimeDurableDate" : ISODate("2018-06-03T05:36:29Z"),

"lastHeartbeat" : ISODate("2018-06-03T05:36:37.754Z"),

"lastHeartbeatRecv" : ISODate("2018-06-03T05:36:38.095Z"),

"pingMs" : NumberLong(1),

"syncingTo" : "192.168.1.113:21000",

"configVersion" : 1

},

{

"\_id" : 2,

"name" : "192.168.1.111:21000",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 242,

"optime" : {

"ts" : Timestamp(1528004189, 2),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1528004189, 2),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T05:36:29Z"),

"optimeDurableDate" : ISODate("2018-06-03T05:36:29Z"),

"lastHeartbeat" : ISODate("2018-06-03T05:36:37.755Z"),

"lastHeartbeatRecv" : ISODate("2018-06-03T05:36:38.200Z"),

"pingMs" : NumberLong(2),

"syncingTo" : "192.168.1.113:21000",

"configVersion" : 1

}

],

"ok" : 1,

"operationTime" : Timestamp(1528004189, 2),

"$gleStats" : {

"lastOpTime" : Timestamp(1528003955, 1),

"electionId" : ObjectId("7fffffff0000000000000001")

},

"$clusterTime" : {

"clusterTime" : Timestamp(1528004189, 2),

"signature" : {

"hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),

"keyId" : NumberLong(0)

}

}

}

7、添加配置文件(三台机器做同样的操作)

cat /etc/mongod/shard1.conf

pidfilepath = /var/run/mongodb/shard1.pid

dbpath = /data/mongodb/shard1/data

logpath = /data/mongodb/shard1/log/shard1.log

logappend = true

bind\_ip = 0.0.0.0

port = 27001

fork = true

#httpinterface=true #打开web监控

#rest=true

replSet=shard1 #副本集名称

shardsvr = true #declare this is a shard db of a cluster;

maxConns=20000 #设置最大连接数

#####################################################

cat /etc/mongod/shard2.conf

pidfilepath = /var/run/mongodb/shard2.pid

dbpath = /data/mongodb/shard2/data

logpath = /data/mongodb/shard2/log/shard2.log

logappend = true

bind\_ip = 0.0.0.0

port = 27002

fork = true

#httpinterface=true #打开web监控

#rest=true

replSet=shard2 #副本集名称

shardsvr = true #declare this is a shard db of a cluster;

maxConns=20000 #设置最大连接数

###################################################

cat /etc/mongod/shard3.conf

pidfilepath = /var/run/mongodb/shard3.pid

dbpath = /data/mongodb/shard3/data

logpath = /data/mongodb/shard3/log/shard3.log

logappend = true

bind\_ip = 0.0.0.0

port = 27001

fork = true

#httpinterface=true #打开web监控

#rest=true

replSet=shard3 #副本集名称

shardsvr = true #declare this is a shard db of a cluster;

maxConns=20000 #设置最大连接数

**8、启动shard1**

111是仲裁

mongod -f /etc/mongod/shard1.conf //启动服务（三台设备做同样操作）

9、登录113或者107任何一台机器的27001端口初始化副本集，111之所以不行，是因为shard1我们把111这台机器的27001端口作为了仲裁节点

在哪台设备上操作，哪台就是主

mongo --port 27001 //在第一台设备上操作或者第二台设备上登录

use admin

config = { \_id: "shard1", members: [ {\_id : 0, host : "192.168.1.113:27001"}, {\_id: 1,host : "192.168.1.107:27001"},{\_id : 2, host : "192.168.1.111:27001",arbiterOnly:true}] }

10、rs.initiate(config)

11、查看状态

rs.status()

{

"set" : "shard1",

"date" : ISODate("2018-06-03T06:27:34.928Z"),

"myState" : 1,

"term" : NumberLong(1),

"heartbeatIntervalMillis" : NumberLong(2000),

"optimes" : {

"lastCommittedOpTime" : {

"ts" : Timestamp(1528007249, 1),

"t" : NumberLong(1)

},

"readConcernMajorityOpTime" : {

"ts" : Timestamp(1528007249, 1),

"t" : NumberLong(1)

},

"appliedOpTime" : {

"ts" : Timestamp(1528007249, 1),

"t" : NumberLong(1)

},

"durableOpTime" : {

"ts" : Timestamp(1528007249, 1),

"t" : NumberLong(1)

}

},

"members" : [

{

"\_id" : 0,

"name" : "192.168.1.113:27001",

"health" : 1,

"state" : 1,

"stateStr" : "PRIMARY",

"uptime" : 1394,

"optime" : {

"ts" : Timestamp(1528007249, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T06:27:29Z"),

"electionTime" : Timestamp(1528007067, 1),

"electionDate" : ISODate("2018-06-03T06:24:27Z"),

"configVersion" : 1,

"self" : true

},

{

"\_id" : 1,

"name" : "192.168.1.107:27001",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 197,

"optime" : {

"ts" : Timestamp(1528007249, 1),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1528007249, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T06:27:29Z"),

"optimeDurableDate" : ISODate("2018-06-03T06:27:29Z"),

"lastHeartbeat" : ISODate("2018-06-03T06:27:33.963Z"),

"lastHeartbeatRecv" : ISODate("2018-06-03T06:27:33.182Z"),

"pingMs" : NumberLong(1),

"syncingTo" : "192.168.1.113:27001",

"configVersion" : 1

},

{

"\_id" : 2,

"name" : "192.168.1.111:27001",

"health" : 1,

"state" : 7,

"stateStr" : "ARBITER",

"uptime" : 197,

"lastHeartbeat" : ISODate("2018-06-03T06:27:33.983Z"),

"lastHeartbeatRecv" : ISODate("2018-06-03T06:27:33.790Z"),

"pingMs" : NumberLong(2),

"configVersion" : 1

}

],

"ok" : 1

}

**9、启动shard2**

113是仲裁

mongod -f /etc/mongod/shard2.conf //三台设备上都要配置

登录107或者111任何一台机器的27002端口初始化副本集，130之所以不行，是因为shard2我们把113这台机器的27002端口作为了仲裁节点

在第二台或者第三台设备上登录，在哪台设备上操作，哪台就是主

mongo --port 27002

use admin

config = { \_id: "shard2", members: [ {\_id : 0, host : "192.168.1.113:27002" ,arbiterOnly:true},{\_id : 1, host : "192.168.1.107:27002"},{\_id : 2, host : "192.168.1.111:27002"}] }

rs.initiate(config)

rs.status()

{

"set" : "shard2",

"date" : ISODate("2018-06-03T15:07:19.490Z"),

"myState" : 1,

"term" : NumberLong(1),

"heartbeatIntervalMillis" : NumberLong(2000),

"optimes" : {

"lastCommittedOpTime" : {

"ts" : Timestamp(1528038436, 1),

"t" : NumberLong(1)

},

"readConcernMajorityOpTime" : {

"ts" : Timestamp(1528038436, 1),

"t" : NumberLong(1)

},

"appliedOpTime" : {

"ts" : Timestamp(1528038436, 1),

"t" : NumberLong(1)

},

"durableOpTime" : {

"ts" : Timestamp(1528038436, 1),

"t" : NumberLong(1)

}

},

"members" : [

{

"\_id" : 0,

"name" : "192.168.1.113:27002",

"health" : 1,

"state" : 7,

"stateStr" : "ARBITER",

"uptime" : 95,

"lastHeartbeat" : ISODate("2018-06-03T15:07:18.921Z"),

"lastHeartbeatRecv" : ISODate("2018-06-03T15:07:17.690Z"),

"pingMs" : NumberLong(1),

"configVersion" : 1

},

{

"\_id" : 1,

"name" : "192.168.1.107:27002",

"health" : 1,

"state" : 1,

"stateStr" : "PRIMARY",

"uptime" : 2008,

"optime" : {

"ts" : Timestamp(1528038436, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T15:07:16Z"),

"infoMessage" : "could not find member to sync from",

"electionTime" : Timestamp(1528038354, 1),

"electionDate" : ISODate("2018-06-03T15:05:54Z"),

"configVersion" : 1,

"self" : true

},

{

"\_id" : 2,

"name" : "192.168.1.111:27002",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 95,

"optime" : {

"ts" : Timestamp(1528038436, 1),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1528038436, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T15:07:16Z"),

"optimeDurableDate" : ISODate("2018-06-03T15:07:16Z"),

"lastHeartbeat" : ISODate("2018-06-03T15:07:18.860Z"),

"lastHeartbeatRecv" : ISODate("2018-06-03T15:07:17.952Z"),

"pingMs" : NumberLong(1),

"syncingTo" : "192.168.1.107:27002",

"configVersion" : 1

}

],

"ok" : 1

}

**10、启动shard3**

mongod -f /etc/mongod/shard3.conf //三台设备均要操作

在第一台（113）或者第三台（111）上操作

107仲裁，在哪台上操作，哪台就被选举成主

config = { \_id: "shard3", members: [ {\_id : 0, host : "192.168.1.113:27003"}, {\_id : 1, host : "192.168.1.107:27003", arbiterOnly:true}, {\_id : 2, host : "192.168.1.111:27003"}] }

rs.initiate(config) //下面是结果

{ "ok" : 1 }

rs.status()

{

"set" : "shard3",

"date" : ISODate("2018-06-03T07:25:52.392Z"),

"myState" : 1,

"term" : NumberLong(1),

"heartbeatIntervalMillis" : NumberLong(2000),

"optimes" : {

"lastCommittedOpTime" : {

"ts" : Timestamp(1528010742, 1),

"t" : NumberLong(1)

},

"readConcernMajorityOpTime" : {

"ts" : Timestamp(1528010742, 1),

"t" : NumberLong(1)

},

"appliedOpTime" : {

"ts" : Timestamp(1528010742, 1),

"t" : NumberLong(1)

},

"durableOpTime" : {

"ts" : Timestamp(1528010742, 1),

"t" : NumberLong(1)

}

},

"members" : [

{

"\_id" : 0,

"name" : "192.168.1.113:27003",

"health" : 1,

"state" : 1,

"stateStr" : "PRIMARY",

"uptime" : 564,

"optime" : {

"ts" : Timestamp(1528010742, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T07:25:42Z"),

"electionTime" : Timestamp(1528010581, 1),

"electionDate" : ISODate("2018-06-03T07:23:01Z"),

"configVersion" : 1,

"self" : true

},

{

"\_id" : 1,

"name" : "192.168.1.107:27003",

"health" : 1,

"state" : 7,

"stateStr" : "ARBITER",

"uptime" : 181,

"lastHeartbeat" : ISODate("2018-06-03T07:25:51.394Z"),

"lastHeartbeatRecv" : ISODate("2018-06-03T07:25:51.241Z"),

"pingMs" : NumberLong(1),

"configVersion" : 1

},

{

"\_id" : 2,

"name" : "192.168.1.111:27003",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 181,

"optime" : {

"ts" : Timestamp(1528010742, 1),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1528010742, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-06-03T07:25:42Z"),

"optimeDurableDate" : ISODate("2018-06-03T07:25:42Z"),

"lastHeartbeat" : ISODate("2018-06-03T07:25:51.467Z"),

"lastHeartbeatRecv" : ISODate("2018-06-03T07:25:52.264Z"),

"pingMs" : NumberLong(2),

"syncingTo" : "192.168.1.113:27003",

"configVersion" : 1

}

],

"ok" : 1

}

**11、添加配置文件(三台机器都操作) 配置mongos**

12、vim /etc/mongod/mongos.conf

pidfilepath = /var/run/mongodb/mongos.pid

logpath = /data/mongodb/mongos/log/mongos.log

logappend = true

bind\_ip = 0.0.0.0

port = 20000

fork = true

configdb = configs/192.168.133.130:21000, 192.168.133.132:21000, 192.168.133.133:21000 #监听的配置服务器,只能有1个或者3个，configs为配置服务器的副本

集名字

maxConns=20000 #设置最大连接数

logappend = true

bind\_ip = 0.0.0.0

port = 20000

fork = true

configdb = configs/192.168.1.113:21000,192.168.1.107:21000,192.168.1.111:21000 #监听的配置服务器,只能有1个或者3个，configs为配置服务器的副本集名字

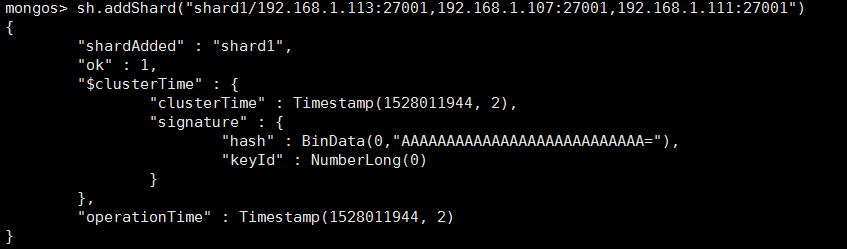
maxConns=20000 #设置最大连接数

13、mongos -f /etc/mongod/mongos.conf //启动服务，三台设备上都要操作

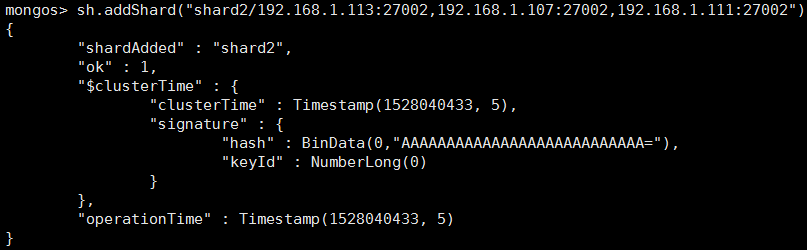
14、登录任何一台20000端口

mongo --port 20000

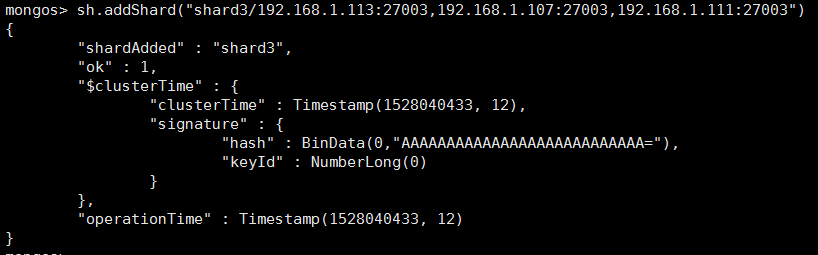
sh.addShard("shard1/192.168.1.113:27001,192.168.1.107:27001,192.168.1.111:27001")



sh.addShard("shard2/192.168.1.113:27002,192.168.1.107:27002,192.168.1.111:27002")



sh.addShard("shard3/192.168.1.113:27003,192.168.1.107:27003,192.168.1.111:27003")



sh.status() //查看状态

--- Sharding Status ---

sharding version: {

"\_id" : 1,

"minCompatibleVersion" : 5,

"currentVersion" : 6,

"clusterId" : ObjectId("5b137d8120594efce633a9ab")

}

shards:

{ "\_id" : "shard1", "host" : "shard1/192.168.1.107:27001,192.168.1.113:27001", "state" : 1 }

{ "\_id" : "shard2", "host" : "shard2/192.168.1.107:27002,192.168.1.111:27002", "state" : 1 }

{ "\_id" : "shard3", "host" : "shard3/192.168.1.111:27003,192.168.1.113:27003", "state" : 1 }

active mongoses:

"3.6.5" : 2

autosplit:

Currently enabled: yes

balancer:

Currently enabled: yes

Currently running: no

Failed balancer rounds in last 5 attempts: 0

Migration Results for the last 24 hours:

No recent migrations

databases:

{ "\_id" : "config", "primary" : "config", "partitioned" : true }

**15、测试**

登录任何一台20000端口

mongo --port 20000

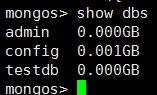
use admin

sh.enableSharding("testdb")

sh.shardCollection("testdb.table1",{"id":1} )

use testdb

for (var i = 1; i <= 10000; i++) db.table1.save({id:i,"test1":"testval1"})



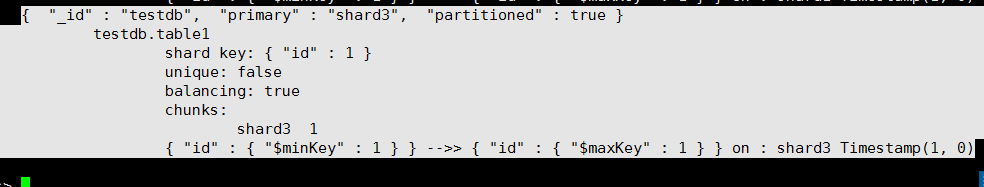
sh.enableSharding("testdb")

sh.shardCollection("testdb.table1",{"id":1} )

use testdb

for (var i = 1; i <= 10000; i++) db.table1.save({id:i,"test1":"testval1"})

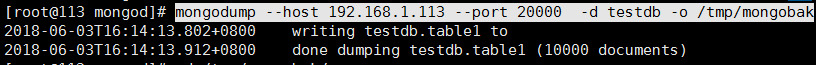
sh.status()



**mongodb备份恢复**

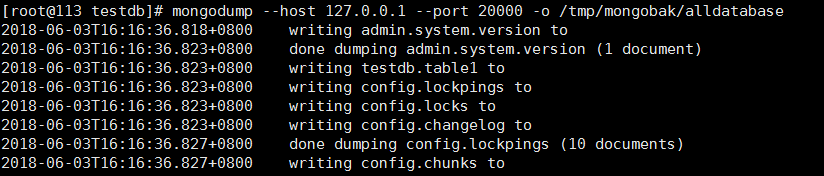
1、mongodump --host 192.168.1.113 --port 20000 -d testdb -o /tmp/mongobak

//--host:指定IP --port：指定端口 -d 备份的数据库名称 -o 备份到设备的位置

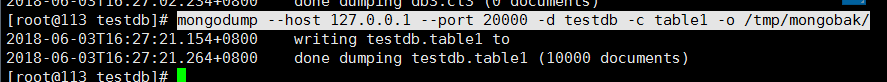


2、备份所有的库

mongodump --host 127.0.0.1 --port 20000 -o /tmp/mongobak/alldatabase/



3、mongodump --host 127.0.0.1 --port 20000 -d testdb -c table1 -o /tmp/mongobak/ //备份指定集合 -c:指定集合名



4、导出集合为json文件

mongoexport --host 127.0.0.1 --port 20000 -d testdb -c table1 -o /tmp/mydb2/1.json

5、恢复所有库

mongo --port 20000

use testdb

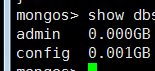
db.dropDatabase() //删除数据库

use db2

db.dropDatabase()

use db3

db.dropDatabase()



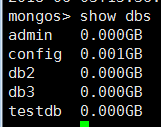
cd /tmp/mongobak/alldatabase //删除不需要恢复的目录，有待测试

rm -rf admin/

rm -rf config/

mongorestore --host 127.0.0.1 --port 20000 /tmp/mongobak/alldatabase//

mongorestore -h 127.0.0.1 --port 20000 --drop dir //其中dir是备份所有库的目录名字，其中--drop可选，意思是当恢复之前先把之前的数据删除，不建议使用



6、恢复指定库

mongorestore --host 127.0.0.1 --port 20000 -d testdb /tmp/mongobak/testdb/

mongorestore -d mydb dir/ //-d跟要恢复的库名字，dir就是该库备份时所在的目录

7、恢复集合

mongorestore -d mydb -c testc dir/mydb/testc.bson // -c后面跟要恢复的集合名字，dir是备份mydb库时生成文件所在路径，这里是一个bson文件的路径

8、导入集合

mongoimport -d mydb -c testc --file /tmp/testc.json