第四十三章：Mongodb分片结合复制集

**一、分片概述；**

**二、分片存储原理；**

**三、分片的片键；**

**四、案例：mongodb分片结合复制集高效存储；**

**一、分片概述：**

**概述：**分片（sharding）是指将数据库拆分，将其分散在不同的机器上的过程。分片集群（sharded cluster）是一种水平扩展数据库系统性能的方法，能够将数据集分布式存储在不同的分片（shard）上，每个分片只保存数据集的一部分，MongoDB保证各个分片之间不会有重复的数据，所有分片保存的数据之和就是完整的数据集。分片集群将数据集分布式存储，能够将负载分摊到多个分片上，每个分片只负责读写一部分数据，充分利用了各个shard的系统资源，提高数据库系统的吞吐量。

注：mongodb3.2版本后，分片技术必须结合复制集完成；

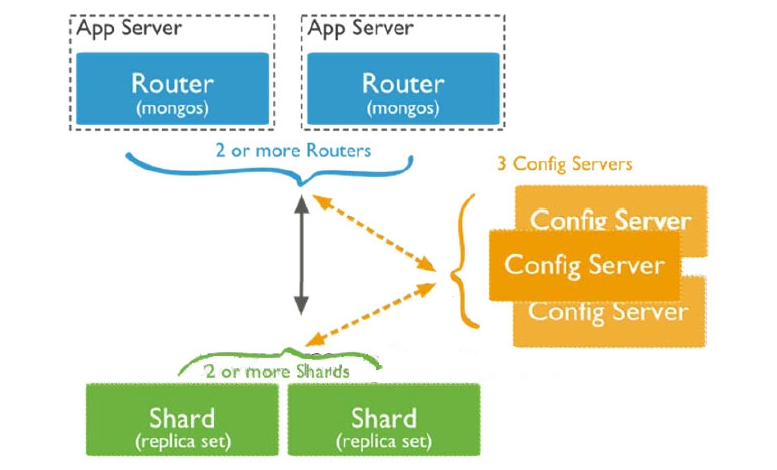
**应用场景：**

1.单台机器的磁盘不够用了，使用分片解决磁盘空间的问题。

2.单个mongod已经不能满足写数据的性能要求。通过分片让写压力分散到各个分片上面，使用分片服务器自身的资源。

3.想把大量数据放到内存里提高性能。和上面一样，通过分片使用分片服务器自身的资源。

**二、分片存储原理：**



**存储方式：**数据集被拆分成数据块（chunk），每个数据块包含多个doc，数据块分布式存储在分片集群中。

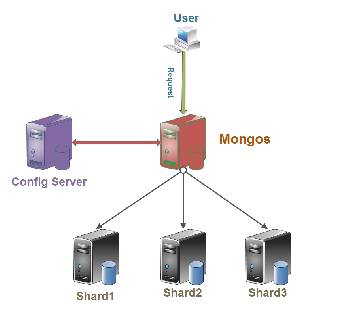
**角色：**

**Config server：**MongoDB负责追踪数据块在shard上的分布信息，每个分片存储哪些数据块，叫做分片的元数据，保存在config server上的数据库 config中，一般使用3台config server，所有config server中的config数据库必须完全相同（建议将config server部署在不同的服务器，以保证稳定性）；

**Shard server：**将数据进行分片，拆分成数据块（chunk），每个trunk块的大小默认为64M，数据块真正存放的单位；

**Mongos server：**数据库集群请求的入口，所有的请求都通过mongos进行协调，查看分片的元数据，查找chunk存放位置，mongos自己就是一个请求分发中心，在生产环境通常有多mongos作为请求的入口，防止其中一个挂掉所有的mongodb请求都没有办法操作。

**总结：**应用请求mongos来操作mongodb的增删改查，配置服务器存储数据库元信息，并且和mongos做同步，数据最终存入在shard（分片）上，为了防止数据丢失，同步在副本集中存储了一份，仲裁节点在数据存储到分片的时候决定存储到哪个节点。



**三、分片的片键；**

**概述：**片键是文档的一个属性字段或是一个复合索引字段，一旦建立后则不可改变，片键是拆分数据的关键的依据，如若在数据极为庞大的场景下，片键决定了数据在分片的过程中数据的存储位置，直接会影响集群的性能；

注：创建片键时，需要有一个支撑片键运行的索引；

**片键分类：**

1.递增片键：使用时间戳，日期，自增的主键，ObjectId，\_id等，此类片键的写入操作集中在一个分片服务器上，写入不具有分散性，这会导致单台服务器压力较大，但分割比较容易，这台服务器可能会成为性能瓶颈；



语法解析：

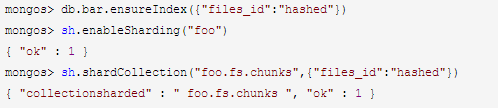
mongos> use 库名

mongos> db.集合名.ensureIndex({"键名":1}) ##创建索引

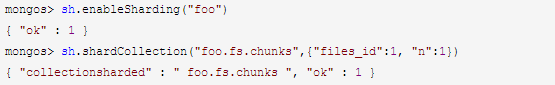
mongos> sh.enableSharding("库名") ##开启库的分片

mongos> sh.shardCollection("库名.集合名",{"键名":1}) ##开启集合的分片并指定片键

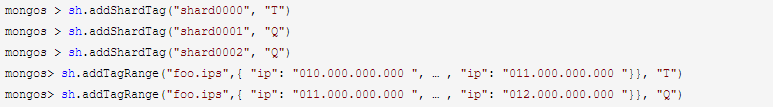
2.哈希片键：也称之为散列索引，使用一个哈希索引字段作为片键，优点是使数据在各节点分布比较均匀，数据写入可随机分发到每个分片服务器上，把写入的压力分散到了各个服务器上。但是读也是随机的，可能会命中更多的分片，但是缺点是无法实现范围区分；



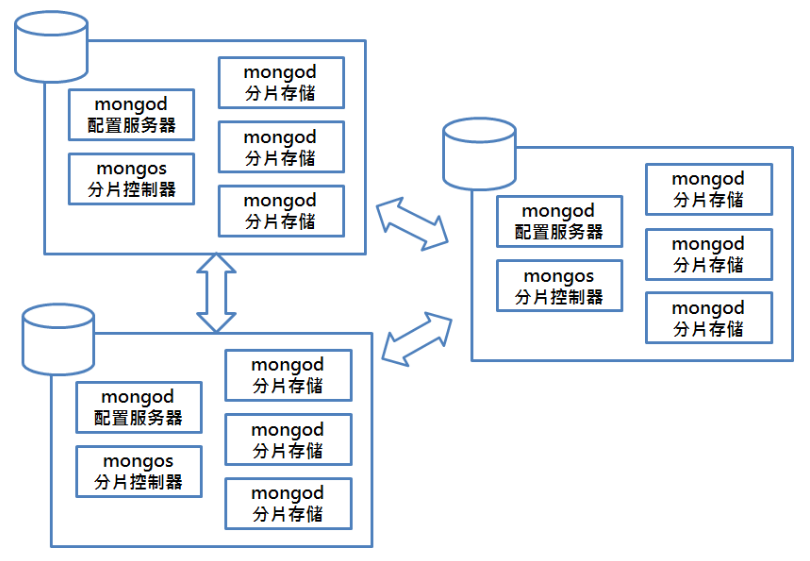
3.组合片键： 数据库中没有比较合适的键值供片键选择，或者是打算使用的片键基数太小（即变化少如星期只有7天可变化），可以选另一个字段使用组合片键，甚至可以添加冗余字段来组合；



4.标签片键：数据存储在指定的分片服务器上，可以为分片添加tag标签，然后指定相应的tag，比如让10.\*.\*.\*(T)出现在shard0000上，11.\*.\*.\*(Q)出现在shard0001或shard0002上，就可以使用tag让均衡器指定分发；



**四、案例：mongodb分片结合复制集高效存储**



**实验环境：**

|  |  |  |
| --- | --- | --- |
| 192.168.100.101  config.linuxfan.cn | 192.168.100.102  shard1.linuxfan.cn | 192.168.100.103  shard2.linuxfan.cn |
| Mongos：27025 | mongos：27025 | mongos：27025 |
| config（configs）：27017 | shard（shard1）：27017 | shard（shard2）：27017 |
| config（configs）：27018 | shard（shard1）：27018 | shard（shard2）：27018 |
| config（configs）：27019 | shard（shard1）：27019 | shard（shard2）：27019 |

**实验步骤：**

* 在所有节点安装mongodb服务：
* 创建config节点的三个实例：
* 配置config节点的configs复制集；
* 配置config节点的mongos进程；
* 创建shard1节点的三个实例：
* 配置shard1节点的shard1复制集；
* 配置shard1节点的mongs进程；
* 创建shard2节点的三个实例：
* 配置shard2节点的shard1复制集；
* 配置shard2节点的mongs进程；
* 选择任意节点的mongos进程配置分片；
* 配置开启testdb数据库和table1集合的分片；
* 创建上述步骤开启的数据库和集合测试是否分片；
* 配置开启testdb2数据库和table1集合的分片；
* 创建上述步骤开启的数据库和集合测试是否分片；
* 扫描某个集合的分片情况；
* 配置开启testdb7数据库和hehe集合的分片（实现在一个集合中的多个document进行分片，通过设置散列片键）；
* 创建上述步骤开启的数据库和集合测试是否分片；
* 扫描某个集合的分片情况进行验证；
* 在192.168.100.102和192.168.100.103上登录mongos节点查看上述配置，发现已经同步；
* 在192.168.100.102和192.168.100.103上登录复制集primary节点查看上述配置，发现已经存在各自的分片；
* 在192.168.100.102上关闭shard1复制集的primary节点，测试mongos访问数据依然没有问题，实现了复制集的高可用；
* **在所有节点安装mongodb服务：**

**192.168.100.101、192.168.100.102、192.168.100.103：**

[root@config ~]# tar zxvf mongodb-linux-x86\_64-rhel70-3.6.3.tgz

[root@config ~]# mv mongodb-linux-x86\_64-rhel70-3.6.3 /usr/local/mongodb

[root@config ~]# echo "export PATH=/usr/local/mongodb/bin:\$PATH" >>/etc/profile

[root@config ~]# source /etc/profile

[root@config ~]# ulimit -n 25000

[root@config ~]# ulimit -u 25000

[root@config ~]# echo 0 >/proc/sys/vm/zone\_reclaim\_mode

[root@config ~]# sysctl -w vm.zone\_reclaim\_mode=0

[root@config ~]# echo never >/sys/kernel/mm/transparent\_hugepage/enabled

[root@config ~]# echo never >/sys/kernel/mm/transparent\_hugepage/defrag

[root@config ~]# cd /usr/local/mongodb/bin/

[root@config bin]# mkdir {../mongodb1,../mongodb2,../mongodb3}

[root@config bin]# mkdir ../logs

[root@config bin]# touch ../logs/mongodb{1..3}.log

[root@config bin]# chmod 777 ../logs/mongodb\*

* **创建config节点的三个实例：**

**192.168.100.101:**

[root@config bin]# cat <<END >>/usr/local/mongodb/bin/mongodb1.conf

bind\_ip=192.168.100.101

port=27017

dbpath=/usr/local/mongodb/mongodb1/

logpath=/usr/local/mongodb/logs/mongodb1.log

logappend=true

fork=true

maxConns=5000

replSet=configs

#replication name

configsvr=true

END

[root@config bin]# cat <<END >>/usr/local/mongodb/bin/mongodb2.conf

bind\_ip=192.168.100.101

port=27018

dbpath=/usr/local/mongodb/mongodb2/

logpath=/usr/local/mongodb/logs/mongodb2.log

logappend=true

fork=true

maxConns=5000

replSet=configs

configsvr=true

END

[root@config bin]# cat <<END >>/usr/local/mongodb/bin/mongodb3.conf

bind\_ip=192.168.100.101

port=27019

dbpath=/usr/local/mongodb/mongodb3/

logpath=/usr/local/mongodb/logs/mongodb3.log

logappend=true

fork=true

maxConns=5000

replSet=configs

configsvr=true

END

[root@config bin]# cd

[root@config ~]# mongod -f /usr/local/mongodb/bin/mongodb1.conf

[root@config ~]# mongod -f /usr/local/mongodb/bin/mongodb2.conf

[root@config ~]# mongod -f /usr/local/mongodb/bin/mongodb3.conf

[root@config ~]# netstat -utpln |grep mongod

tcp 0 0 192.168.100.101:27019 0.0.0.0:\* LISTEN 2271/mongod

tcp 0 0 192.168.100.101:27017 0.0.0.0:\* LISTEN 2440/mongod

tcp 0 0 192.168.100.101:27018 0.0.0.0:\* LISTEN 1412/mongod

[root@config ~]# echo -e "/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/mongodb1.conf \n/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/mongodb2.conf\n/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/mongodb3.conf">>/etc/rc.local

[root@config ~]# chmod +x /etc/rc.local

[root@config ~]# cat <<END >>/etc/init.d/mongodb

#!/bin/bash

INSTANCE=\$1

ACTION=\$2

case "\$ACTION" in

'start')

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf;;

'stop')

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf --shutdown;;

'restart')

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf --shutdown

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf;;

esac

END

[root@config ~]# chmod +x /etc/init.d/mongodb

* **配置config节点的configs复制集；**

[root@config ~]# mongo --port 27017 --host 192.168.100.101

> cfg={"\_id":"configs","members":[{"\_id":0,"host":"192.168.100.101:27017"},{"\_id":1,"host":"192.168.100.101:27018"},{"\_id":2,"host":"192.168.100.101:27019"}]}

> rs.initiate(cfg)

configs:PRIMARY> rs.status()

{

"set" : "configs",

"date" : ISODate("2018-04-24T18:53:44.375Z"),

"myState" : 1,

"term" : NumberLong(1),

"configsvr" : true,

"heartbeatIntervalMillis" : NumberLong(2000),

"optimes" : {

"lastCommittedOpTime" : {

"ts" : Timestamp(1524596020, 1),

"t" : NumberLong(1)

},

"readConcernMajorityOpTime" : {

"ts" : Timestamp(1524596020, 1),

"t" : NumberLong(1)

},

"appliedOpTime" : {

"ts" : Timestamp(1524596020, 1),

"t" : NumberLong(1)

},

"durableOpTime" : {

"ts" : Timestamp(1524596020, 1),

"t" : NumberLong(1)

}

},

"members" : [

{

"\_id" : 0,

"name" : "192.168.100.101:27017",

"health" : 1,

"state" : 1,

"stateStr" : "PRIMARY",

"uptime" : 6698,

"optime" : {

"ts" : Timestamp(1524596020, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-04-24T18:53:40Z"),

"electionTime" : Timestamp(1524590293, 1),

"electionDate" : ISODate("2018-04-24T17:18:13Z"),

"configVersion" : 1,

"self" : true

},

{

"\_id" : 1,

"name" : "192.168.100.101:27018",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 5741,

"optime" : {

"ts" : Timestamp(1524596020, 1),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1524596020, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-04-24T18:53:40Z"),

"optimeDurableDate" : ISODate("2018-04-24T18:53:40Z"),

"lastHeartbeat" : ISODate("2018-04-24T18:53:42.992Z"),

"lastHeartbeatRecv" : ISODate("2018-04-24T18:53:43.742Z"),

"pingMs" : NumberLong(0),

"syncingTo" : "192.168.100.101:27017",

"configVersion" : 1

},

{

"\_id" : 2,

"name" : "192.168.100.101:27019",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 5741,

"optime" : {

"ts" : Timestamp(1524596020, 1),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1524596020, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-04-24T18:53:40Z"),

"optimeDurableDate" : ISODate("2018-04-24T18:53:40Z"),

"lastHeartbeat" : ISODate("2018-04-24T18:53:42.992Z"),

"lastHeartbeatRecv" : ISODate("2018-04-24T18:53:43.710Z"),

"pingMs" : NumberLong(0),

"syncingTo" : "192.168.100.101:27017",

"configVersion" : 1

}

],

"ok" : 1,

"operationTime" : Timestamp(1524596020, 1),

"$gleStats" : {

"lastOpTime" : Timestamp(0, 0),

"electionId" : ObjectId("7fffffff0000000000000001")

},

"$clusterTime" : {

"clusterTime" : Timestamp(1524596020, 1),

"signature" : {

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

"keyId" : NumberLong(0)

}

}

}

configs:PRIMARY> show dbs

admin 0.000GB

config 0.000GB

local 0.000GB

configs:PRIMARY> exit

* **配置config节点的mongos进程；**

[root@config bin]# cat <<END >>/usr/local/mongodb/bin/mongos.conf

bind\_ip=192.168.100.101

port=27025

logpath=/usr/local/mongodb/logs/mongos.log

fork=true

maxConns=5000

configdb=configs/192.168.100.101:27017,192.168.100.101:27018,192.168.100.101:27019

END

注：mongos的configdb参数只能指定一个（复制集中的primary）或多个（复制集中的全部节点）；

[root@config bin]# touch ../logs/mongos.log

[root@config bin]# chmod 777 ../logs/mongos.log

[root@config bin]# mongos -f /usr/local/mongodb/bin/mongos.conf

about to fork child process, waiting until server is ready for connections.

forked process: 1562

child process started successfully, parent exiting

[root@config ~]# netstat -utpln |grep mongo

tcp 0 0 192.168.100.101:27019 0.0.0.0:\* LISTEN 1601/mongod

tcp 0 0 192.168.100.101:27020 0.0.0.0:\* LISTEN 1345/mongod

tcp 0 0 192.168.100.101:27025 0.0.0.0:\* LISTEN 1822/mongos

tcp 0 0 192.168.100.101:27017 0.0.0.0:\* LISTEN 1437/mongod

tcp 0 0 192.168.100.101:27018 0.0.0.0:\* LISTEN 1541/mongod

* **创建shard1节点的三个实例：**

**192.168.100.102:**

[root@shard1 bin]# cat <<END >>/usr/local/mongodb/bin/mongodb1.conf

bind\_ip=192.168.100.102

port=27017

dbpath=/usr/local/mongodb/mongodb1/

logpath=/usr/local/mongodb/logs/mongodb1.log

logappend=true

fork=true

maxConns=5000

replSet=shard1

#replication name

shardsvr=true

END

[root@shard1 bin]# cat <<END >>/usr/local/mongodb/bin/mongodb2.conf

bind\_ip=192.168.100.102

port=27018

dbpath=/usr/local/mongodb/mongodb2/

logpath=/usr/local/mongodb/logs/mongodb2.log

logappend=true

fork=true

maxConns=5000

replSet=shard1

shardsvr=true

END

[root@shard1 bin]# cat <<END >>/usr/local/mongodb/bin/mongodb3.conf

bind\_ip=192.168.100.102

port=27019

dbpath=/usr/local/mongodb/mongodb3/

logpath=/usr/local/mongodb/logs/mongodb3.log

logappend=true

fork=true

maxConns=5000

replSet=shard1

shardsvr=true

END

[root@shard1 bin]# cd

[root@shard1 ~]# mongod -f /usr/local/mongodb/bin/mongodb1.conf

[root@shard1 ~]# mongod -f /usr/local/mongodb/bin/mongodb2.conf

[root@shard1 ~]# mongod -f /usr/local/mongodb/bin/mongodb3.conf

[root@shard1 ~]# netstat -utpln |grep mongod

tcp 0 0 192.168.100.101:27019 0.0.0.0:\* LISTEN 2271/mongod

tcp 0 0 192.168.100.101:27017 0.0.0.0:\* LISTEN 2440/mongod

tcp 0 0 192.168.100.101:27018 0.0.0.0:\* LISTEN 1412/mongod

[root@shard1 ~]# echo -e "/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/mongodb1.conf \n/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/mongodb2.conf\n/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/mongodb3.conf">>/etc/rc.local

[root@shard1 ~]# chmod +x /etc/rc.local

[root@shard1 ~]# cat <<END >>/etc/init.d/mongodb

#!/bin/bash

INSTANCE=\$1

ACTION=\$2

case "\$ACTION" in

'start')

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf;;

'stop')

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf --shutdown;;

'restart')

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf --shutdown

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf;;

esac

END

[root@shard1 ~]# chmod +x /etc/init.d/mongodb

* **配置shard1节点的shard1复制集；**

[root@shard1 ~]# mongo --port 27017 --host 192.168.100.102

>cfg={"\_id":"shard1","members":[{"\_id":0,"host":"192.168.100.102:27017"},{"\_id":1,"host":"192.168.100.102:27018"},{"\_id":2,"host":"192.168.100.102:27019"}]}

> rs.initiate(cfg)

{ "ok" : 1 }

shard1:PRIMARY> rs.status()

{

"set" : "shard1",

"date" : ISODate("2018-04-24T19:06:53.160Z"),

"myState" : 1,

"term" : NumberLong(1),

"heartbeatIntervalMillis" : NumberLong(2000),

"optimes" : {

"lastCommittedOpTime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"readConcernMajorityOpTime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"appliedOpTime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"durableOpTime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

}

},

"members" : [

{

"\_id" : 0,

"name" : "192.168.100.102:27017",

"health" : 1,

"state" : 1,

"stateStr" : "PRIMARY",

"uptime" : 6648,

"optime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-04-24T19:06:50Z"),

"electionTime" : Timestamp(1524590628, 1),

"electionDate" : ISODate("2018-04-24T17:23:48Z"),

"configVersion" : 1,

"self" : true

},

{

"\_id" : 1,

"name" : "192.168.100.102:27018",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 6195,

"optime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-04-24T19:06:50Z"),

"optimeDurableDate" : ISODate("2018-04-24T19:06:50Z"),

"lastHeartbeat" : ISODate("2018-04-24T19:06:52.176Z"),

"lastHeartbeatRecv" : ISODate("2018-04-24T19:06:52.626Z"),

"pingMs" : NumberLong(0),

"syncingTo" : "192.168.100.102:27017",

"configVersion" : 1

},

{

"\_id" : 2,

"name" : "192.168.100.102:27019",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 6195,

"optime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-04-24T19:06:50Z"),

"optimeDurableDate" : ISODate("2018-04-24T19:06:50Z"),

"lastHeartbeat" : ISODate("2018-04-24T19:06:52.177Z"),

"lastHeartbeatRecv" : ISODate("2018-04-24T19:06:52.626Z"),

"pingMs" : NumberLong(0),

"syncingTo" : "192.168.100.102:27017",

"configVersion" : 1

}

],

"ok" : 1

}

shard1:PRIMARY> show dbs

admin 0.000GB

config 0.000GB

local 0.000GB

shard1:PRIMARY> exit

* **配置shard1节点的mongs进程；**

[root@shard1 bin]# cat <<END >>/usr/local/mongodb/bin/mongos.conf

bind\_ip=192.168.100.102

port=27025

logpath=/usr/local/mongodb/logs/mongos.log

fork=true

maxConns=5000

configdb=configs/192.168.100.101:27017,192.168.100.101:27018,192.168.100.101:27019

END

[root@shard1 bin]# touch ../logs/mongos.log

[root@shard1 bin]# chmod 777 ../logs/mongos.log

[root@shard1 bin]# mongos -f /usr/local/mongodb/bin/mongos.conf

about to fork child process, waiting until server is ready for connections.

forked process: 1562

child process started successfully, parent exiting

[root@shard1 ~]# netstat -utpln| grep mongo

tcp 0 0 192.168.100.102:27019 0.0.0.0:\* LISTEN 1098/mongod

tcp 0 0 192.168.100.102:27020 0.0.0.0:\* LISTEN 1125/mongod

tcp 0 0 192.168.100.102:27025 0.0.0.0:\* LISTEN 1562/mongos

tcp 0 0 192.168.100.102:27017 0.0.0.0:\* LISTEN 1044/mongod

tcp 0 0 192.168.100.102:27018 0.0.0.0:\* LISTEN 1071/mongod

* **创建shard2节点的三个实例：**

**192.168.100.103:**

[root@shard2 bin]# cat <<END >>/usr/local/mongodb/bin/mongodb1.conf

bind\_ip=192.168.100.103

port=27017

dbpath=/usr/local/mongodb/mongodb1/

logpath=/usr/local/mongodb/logs/mongodb1.log

logappend=true

fork=true

maxConns=5000

replSet=shard2

#replication name

shardsvr=true

END

[root@shard2 bin]# cat <<END >>/usr/local/mongodb/bin/mongodb2.conf

bind\_ip=192.168.100.103

port=27018

dbpath=/usr/local/mongodb/mongodb2/

logpath=/usr/local/mongodb/logs/mongodb2.log

logappend=true

fork=true

maxConns=5000

replSet=shard2

shardsvr=true

END

[root@shard2 bin]# cat <<END >>/usr/local/mongodb/bin/mongodb3.conf

bind\_ip=192.168.100.103

port=27019

dbpath=/usr/local/mongodb/mongodb3/

logpath=/usr/local/mongodb/logs/mongodb3.log

logappend=true

fork=true

maxConns=5000

replSet=shard2

shardsvr=true

END

[root@shard2 bin]# cd

[root@shard2 ~]# mongod -f /usr/local/mongodb/bin/mongodb1.conf

[root@shard2 ~]# mongod -f /usr/local/mongodb/bin/mongodb2.conf

[root@shard2 ~]# mongod -f /usr/local/mongodb/bin/mongodb3.conf

[root@shard2 ~]# netstat -utpln |grep mongod

tcp 0 0 192.168.100.101:27019 0.0.0.0:\* LISTEN 2271/mongod

tcp 0 0 192.168.100.101:27017 0.0.0.0:\* LISTEN 2440/mongod

tcp 0 0 192.168.100.101:27018 0.0.0.0:\* LISTEN 1412/mongod

[root@shard2 ~]# echo -e "/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/mongodb1.conf \n/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/mongodb2.conf\n/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/mongodb3.conf">>/etc/rc.local

[root@shard2 ~]# chmod +x /etc/rc.local

[root@shard2 ~]# cat <<END >>/etc/init.d/mongodb

#!/bin/bash

INSTANCE=\$1

ACTION=\$2

case "\$ACTION" in

'start')

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf;;

'stop')

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf --shutdown;;

'restart')

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf --shutdown

/usr/local/mongodb/bin/mongod -f /usr/local/mongodb/bin/"\$INSTANCE".conf;;

esac

END

[root@shard2 ~]# chmod +x /etc/init.d/mongodb

* **配置shard2节点的shard2复制集；**

[root@shard2 ~]# mongo --port 27017 --host 192.168.100.103

>cfg={"\_id":"shard2","members":[{"\_id":0,"host":"192.168.100.103:27017"},{"\_id":1,"host":"192.168.100.103:27018"},{"\_id":2,"host":"192.168.100.103:27019"}]}

> rs.initiate(cfg)

{ "ok" : 1 }

shard2:PRIMARY> rs.status()

{

"set" : "shard2",

"date" : ISODate("2018-04-24T19:06:53.160Z"),

"myState" : 1,

"term" : NumberLong(1),

"heartbeatIntervalMillis" : NumberLong(2000),

"optimes" : {

"lastCommittedOpTime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"readConcernMajorityOpTime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"appliedOpTime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"durableOpTime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

}

},

"members" : [

{

"\_id" : 0,

"name" : "192.168.100.103:27017",

"health" : 1,

"state" : 1,

"stateStr" : "PRIMARY",

"uptime" : 6648,

"optime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-04-24T19:06:50Z"),

"electionTime" : Timestamp(1524590628, 1),

"electionDate" : ISODate("2018-04-24T17:23:48Z"),

"configVersion" : 1,

"self" : true

},

{

"\_id" : 1,

"name" : "192.168.100.103:27018",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 6195,

"optime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-04-24T19:06:50Z"),

"optimeDurableDate" : ISODate("2018-04-24T19:06:50Z"),

"lastHeartbeat" : ISODate("2018-04-24T19:06:52.176Z"),

"lastHeartbeatRecv" : ISODate("2018-04-24T19:06:52.626Z"),

"pingMs" : NumberLong(0),

"syncingTo" : "192.168.100.103:27017",

"configVersion" : 1

},

{

"\_id" : 2,

"name" : "192.168.100.103:27019",

"health" : 1,

"state" : 2,

"stateStr" : "SECONDARY",

"uptime" : 6195,

"optime" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"optimeDurable" : {

"ts" : Timestamp(1524596810, 1),

"t" : NumberLong(1)

},

"optimeDate" : ISODate("2018-04-24T19:06:50Z"),

"optimeDurableDate" : ISODate("2018-04-24T19:06:50Z"),

"lastHeartbeat" : ISODate("2018-04-24T19:06:52.177Z"),

"lastHeartbeatRecv" : ISODate("2018-04-24T19:06:52.626Z"),

"pingMs" : NumberLong(0),

"syncingTo" : "192.168.100.103:27017",

"configVersion" : 1

}

],

"ok" : 1

}

shard2:PRIMARY> show dbs

admin 0.000GB

config 0.000GB

local 0.000GB

shard2:PRIMARY> exit

* **配置shard2节点的mongs进程；**

[root@shard2 bin]# cat <<END >>/usr/local/mongodb/bin/mongos.conf

bind\_ip=192.168.100.103

port=27025

logpath=/usr/local/mongodb/logs/mongos.log

fork=true

maxConns=5000

configdb=configs/192.168.100.101:27017,192.168.100.101:27018,192.168.100.101:27019

END

[root@shard2 bin]# touch ../logs/mongos.log

[root@shard2 bin]# chmod 777 ../logs/mongos.log

[root@shard2 bin]# mongos -f /usr/local/mongodb/bin/mongos.conf

about to fork child process, waiting until server is ready for connections.

forked process: 1562

child process started successfully, parent exiting

[root@shard2 ~]# netstat -utpln |grep mongo

tcp 0 0 192.168.100.103:27019 0.0.0.0:\* LISTEN 1095/mongod

tcp 0 0 192.168.100.103:27020 0.0.0.0:\* LISTEN 1122/mongod

tcp 0 0 192.168.100.103:27025 0.0.0.0:\* LISTEN 12122/mongos

tcp 0 0 192.168.100.103:27017 0.0.0.0:\* LISTEN 1041/mongod

tcp 0 0 192.168.100.103:27018 0.0.0.0:\* LISTEN 1068/mongod

* **选择任意节点的mongos进程配置分片；**

**192.168.100.101(随意选择mongos进行设置分片，三台mongos会同步任何一台的操作)：**

[root@config ~]# mongo --port 27025 --host 192.168.100.101

mongos> use admin;

switched to db admin

mongos> sh.status() ##查看分片的状态

--- Sharding Status ---

sharding version: {

"\_id" : 1,

"minCompatibleVersion" : 5,

"currentVersion" : 6,

"clusterId" : ObjectId("5adf66d7518b3e5b3aad4e77")

}

shards:

active mongoses:

"3.6.3" : 1

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 }

mongos>

sh.addShard("shard1/192.168.100.102:27017,192.168.100.102:27018,192.168.100.102:27019")

##创建shard1第一个分片

{

"shardAdded" : "shard1",

"ok" : 1,

"$clusterTime" : {

"clusterTime" : Timestamp(1524598580, 9),

"signature" : {

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

"keyId" : NumberLong(0)

}

},

"operationTime" : Timestamp(1524598580, 9)

}

mongos> sh.addShard("shard2/192.168.100.103:27017,192.168.100.103:27018,192.168.100.103:27019")

##创建shard2第二个分片

{

"shardAdded" : "shard2",

"ok" : 1,

"$clusterTime" : {

"clusterTime" : Timestamp(1524598657, 7),

"signature" : {

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

"keyId" : NumberLong(0)

}

},

"operationTime" : Timestamp(1524598657, 7)

}

mongos> sh.status()

--- Sharding Status ---

sharding version: {

"\_id" : 1,

"minCompatibleVersion" : 5,

"currentVersion" : 6,

"clusterId" : ObjectId("5adf66d7518b3e5b3aad4e77")

}

shards:

{ "\_id" : "shard1", "host" : "shard1/192.168.100.102:27017,192.168.100.102:27018,192.168.100.102:27019", "state" : 1 }

{ "\_id" : "shard2", "host" : "shard2/192.168.100.103:27017,192.168.100.103:27018,192.168.100.103:27019", "state" : 1 }

active mongoses:

"3.6.3" : 1

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 }

注：目前配置服务、路由服务、分片服务、副本集服务都已经串联起来了，但我们的目的是希望插入数据，数据能够自动分片。连接在mongos上，准备让指定的数据库、指定的集合分片生效。

注：configs复制集内无法创建数据，shard1和shard2复制集内可以创建数据；

* **配置开启testdb数据库和table1集合的分片；**

[root@config ~]# mongo --port 27025 --host 192.168.100.101

mongos> use admin

mongos> sh.enableSharding("testdb") ##开启数据库的分片

{

"ok" : 1,

"$clusterTime" : {

"clusterTime" : Timestamp(1524599672, 13),

"signature" : {

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

"keyId" : NumberLong(0)

}

},

"operationTime" : Timestamp(1524599672, 13)

mongos> sh.status()

--- Sharding Status ---

sharding version: {

"\_id" : 1,

"minCompatibleVersion" : 5,

"currentVersion" : 6,

"clusterId" : ObjectId("5adf66d7518b3e5b3aad4e77")

}

shards:

{ "\_id" : "shard1", "host" : "shard1/192.168.100.102:27017,192.168.100.102:27018,192.168.100.102:27019", "state" : 1 }

{ "\_id" : "shard2", "host" : "shard2/192.168.100.103:27017,192.168.100.103:27018,192.168.100.103:27019", "state" : 1 }

active mongoses:

"3.6.3" : 1

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 }

config.system.sessions

shard key: { "\_id" : 1 }

unique: false

balancing: true

chunks:

shard1 1

{ "\_id" : { "$minKey" : 1 } } -->> { "\_id" : { "$maxKey" : 1 } } on : shard1 Timestamp(1, 0)

{ "\_id" : "testdb", "primary" : "shard2", "partitioned" : true }

mongos> db.runCommand({shardcollection:"testdb.table1", key:{\_id:1}}); ##开启数据库中集合的分片，同时会创建该数据库该集合

{

"collectionsharded" : "testdb.table1",

"collectionUUID" : UUID("883bb1e2-b218-41ab-8122-6a5cf4df5e7b"),

"ok" : 1,

"$clusterTime" : {

"clusterTime" : Timestamp(1524601471, 14),

"signature" : {

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

"keyId" : NumberLong(0)

}

},

"operationTime" : Timestamp(1524601471, 14)

}

* **创建上述步骤开启的数据库和集合测试是否分片；**

[root@config ~]# mongo --port 27025 --host 192.168.100.101

mongos> use testdb; ##创建数据库中的记录，测试分片情况；

mongos> for(i=1;i<=10000;i++){db.table1.insert({"id":i,"name":"huge"})};

WriteResult({ "nInserted" : 1 })

mongos> show collections

table1

mongos> db.table1.count()

10000

mongos> sh.status()

--- Sharding Status ---

sharding version: {

"\_id" : 1,

"minCompatibleVersion" : 5,

"currentVersion" : 6,

"clusterId" : ObjectId("5adf66d7518b3e5b3aad4e77")

}

shards:

{ "\_id" : "shard1", "host" : "shard1/192.168.100.102:27017,192.168.100.102:27018,192.168.100.102:27019", "state" : 1 }

{ "\_id" : "shard2", "host" : "shard2/192.168.100.103:27017,192.168.100.103:27018,192.168.100.103:27019", "state" : 1 }

active mongoses:

"3.6.3" : 1

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 }

config.system.sessions

shard key: { "\_id" : 1 }

unique: false

balancing: true

chunks:

shard1 1

{ "\_id" : { "$minKey" : 1 } } -->> { "\_id" : { "$maxKey" : 1 } } on : shard1 Timestamp(1, 0)

{ "\_id" : "testdb", "primary" : "shard2", "partitioned" : true }

testdb.table1

shard key: { "\_id" : 1 }

unique: false

balancing: true

chunks:

shard2 1

{ "\_id" : { "$minKey" : 1 } } -->> { "\_id" : { "$maxKey" : 1 } } on : shard2 Timestamp(1, 0)

* **配置开启testdb2数据库和table1的分片；**

[root@config ~]# mongo --port 27025 --host 192.168.100.101

mongos> use admin ##开启testdb2的分片

switched to db admin

mongos> sh.enableSharding("testdb2")

{

"ok" : 1,

"$clusterTime" : {

"clusterTime" : Timestamp(1524602371, 7),

"signature" : {

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

"keyId" : NumberLong(0)

}

},

"operationTime" : Timestamp(1524602371, 7)

}

mongos> db.runCommand({shardcollection:"testdb2.table1", key:{\_id:1}});

##开启testdb2下table1集合的分片

* **创建上述步骤开启的数据库和集合测试是否分片；**

[root@config ~]# mongo --port 27025 --host 192.168.100.101

mongos> use testdb2 ##创建数据进行测试testdb2.table1的分片情况

switched to db testdb2

mongos> for(i=1;i<=10000;i++){db.table1.insert({"id":i,"name":"huge"})};

WriteResult({ "nInserted" : 1 })

mongos> sh.status()

--- Sharding Status ---

sharding version: {

"\_id" : 1,

"minCompatibleVersion" : 5,

"currentVersion" : 6,

"clusterId" : ObjectId("5adf66d7518b3e5b3aad4e77")

}

shards:

{ "\_id" : "shard1", "host" : "shard1/192.168.100.102:27017,192.168.100.102:27018,192.168.100.102:27019", "state" : 1 }

{ "\_id" : "shard2", "host" : "shard2/192.168.100.103:27017,192.168.100.103:27018,192.168.100.103:27019", "state" : 1 }

active mongoses:

"3.6.3" : 1

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 }

config.system.sessions

shard key: { "\_id" : 1 }

unique: false

balancing: true

chunks:

shard1 1

{ "\_id" : { "$minKey" : 1 } } -->> { "\_id" : { "$maxKey" : 1 } } on : shard1 Timestamp(1, 0)

{ "\_id" : "testdb", "primary" : "shard2", "partitioned" : true }

testdb.table1

shard key: { "\_id" : 1 }

unique: false

balancing: true

chunks:

shard2 1

{ "\_id" : { "$minKey" : 1 } } -->> { "\_id" : { "$maxKey" : 1 } } on : shard2 Timestamp(1, 0)

{ "\_id" : "testdb2", "primary" : "shard1", "partitioned" : true }

testdb2.table1

shard key: { "\_id" : 1 }

unique: false

balancing: true

chunks:

shard1 1

{ "\_id" : { "$minKey" : 1 } } -->> { "\_id" : { "$maxKey" : 1 } } on : shard1 Timestamp(1, 0)

* **扫描某个集合的分片情况；**

[root@config ~]# mongo --port 27025 --host 192.168.100.101

mongos> use testdb2

switched to db testdb2

mongos> db.table1.stats() ##查看集合的分片情况

{

"sharded" : true,

"capped" : false,

"ns" : "testdb2.table1",

"count" : 10000,

"size" : 490000,

"storageSize" : 167936,

"totalIndexSize" : 102400,

"indexSizes" : {

"\_id\_" : 102400

},

"avgObjSize" : 49,

"nindexes" : 1,

"nchunks" : 1,

"shards" : {

"shard1" : {

"ns" : "testdb2.table1",

"size" : 490000,

"count" : 10000,

"avgObjSize" : 49,

"storageSize" : 167936,

"capped" : false,

"wiredTiger" : {

"metadata" : {

"formatVersion" : 1

},

"creationString" :

...

* **配置开启testdb7数据库和hehe集合的分片（实现在一个集合中的多个document进行分片，通过设置散列片键）；**

mongos> db.hehe.ensureIndex({"id":"hashed"}) ##创建hash索引，名称为hehe，以此作为散列片键分片使用，设置键名为id，作为分片的依据

{

"raw" : {

"shard2/192.168.100.103:27017,192.168.100.103:27018,192.168.100.103:27019" : {

"createdCollectionAutomatically" : true,

"numIndexesBefore" : 1,

"numIndexesAfter" : 2,

"ok" : 1

}

},

"ok" : 1,

"$clusterTime" : {

"clusterTime" : Timestamp(1534192213, 2),

"signature" : {

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

"keyId" : NumberLong(0)

}

},

"operationTime" : Timestamp(1534192213, 2)

}

mongos> sh.enableSharding("testdb7"); ##开启testdb7数据库的分片

{

"ok" : 1,

"$clusterTime" : {

"clusterTime" : Timestamp(1534192235, 6),

"signature" : {

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

"keyId" : NumberLong(0)

}

},

"operationTime" : Timestamp(1534192235, 6)

}

mongos> sh.shardCollection("testdb7.hehe",{"id":"hashed"}) ##开启数据库中hehe集合的分片，采用id键名作为分片依据；

{

"collectionsharded" : "testdb7.hehe",

"collectionUUID" : UUID("03ae881d-9cd2-4445-81a4-781ec9bacc44"),

"ok" : 1,

"$clusterTime" : {

"clusterTime" : Timestamp(1534192301, 20),

"signature" : {

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

"keyId" : NumberLong(0)

}

},

"operationTime" : Timestamp(1534192301, 12)

}

* **创建上述步骤开启的数据库和集合测试是否分片；**

mongos> use testdb7

switched to db testdb7

mongos> for(i=1;i<=1000;i++){db.hehe.insert({"id":i,"name":"huge"})};

WriteResult({ "nInserted" : 1 })

* **扫描某个集合的分片情况进行验证；**

mongos> db.hehe.stats()

{

"sharded" : true,

"capped" : false,

"ns" : "testdb7.hehe",

"count" : 1000,

"size" : 49000,

"storageSize" : 40960,

"totalIndexSize" : 73728,

"indexSizes" : {

"\_id\_" : 32768,

"id\_hashed" : 40960

},

"avgObjSize" : 49,

"nindexes" : 2,

"nchunks" : 4,

"shards" : {

"shard1" : {

"ns" : "testdb7.hehe",

"size" : 23520,

"count" : 480,

"avgObjSize" : 49,

"storageSize" : 20480,

"capped" : false,

"wiredTiger" : {

"metadata" : {

"formatVersion" : 1

},

...

"shard2" : {

"ns" : "testdb7.hehe",

"size" : 25480,

"count" : 520,

"avgObjSize" : 49,

"storageSize" : 20480,

"capped" : false,

"wiredTiger" : {

"metadata" : {

"formatVersion" : 1

},

"creationString" :

...

* **在192.168.100.102和192.168.100.103上登录mongos节点查看上述配置，发现已经同步；**
* **在192.168.100.102和192.168.100.103上登录复制集primary节点查看上述配置，发现已经存在各自的分片；**
* **在192.168.100.102上关闭shard1复制集的primary节点，测试mongos访问数据依然没有问题，实现了复制集的高可用；**