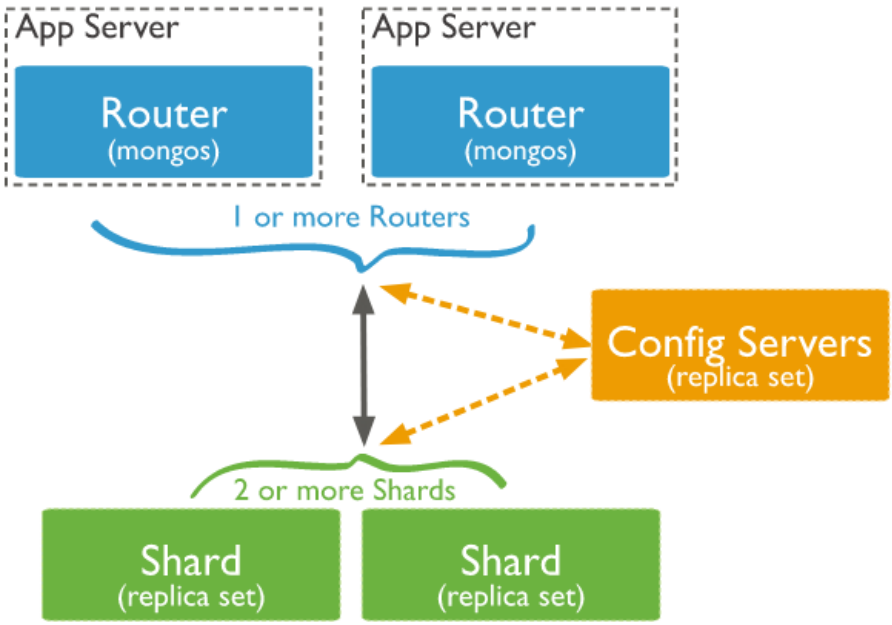
MongoDB 3.4 分片 + 副本集 集群搭建

mongodb是最常用的nodql数据库，2017年6月，在数据库排名中已经上升到了第五。这篇文章介绍如何搭建高可用的mongodb（分片+副本）集群。

在搭建集群之前，需要首先了解几个概念：路由，分片、副本集、配置服务器等。



图中可以看到有四个组件：mongos、config server、shard、replica set

mongos：数据库集群请求的入口，所有的请求都通过mongos进行协调，不需要在应用程序添加一个路由选择器，mongos自己就是一个请求分发中心，它负责把对应的数据请求请求转发到对应的shard服务器上。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。

config server：为配置服务器，存储所有数据库元信息（路由、分片）的配置。mongos本身没有物理存储分片服务器和数据路由信息，只是缓存在内存里，配置服务器则实际存储这些数据。mongos第一次启动或者关掉重启就会从 config server 加载配置信息，以后如果配置服务器信息变化会通知到所有的 mongos 更新自己的状态，这样 mongos 就能继续准确路由。在生产环境通常有多个 config server 配置服务器，因为它存储了分片路由的元数据，防止数据丢失！在MongoDB 3.4 中要求config server 必须是一个副本集

shard：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。将数据分散到不同的机器上，不需要功能强大的服务器就可以存储更多的数据和处理更大的负载。基本思想就是将集合切成小块，这些块分散到若干片里，每个片只负责总数据的一部分，最后通过一个均衡器来对各个分片进行均衡（数据迁移）。

replica set：副本集，其实就是shard的备份，防止shard挂掉之后数据丢失。复制提供了数据的冗余备份，并在多个服务器上存储数据副本，提高了数据的可用性， 并可以保证数据的安全性。

仲裁者（Arbiter），是复制集中的一个MongoDB实例，它并不保存数据。仲裁节点使用最小的资源并且不

要求硬件设备，不能将Arbiter部署在同一个数据集节点中，可以部署在其他应用服务器或者监视服务器中，也可部署在单独的虚拟机中。为了确保复制集中有奇数的投票成员（包括primary），需要添加仲裁节点做为投票，否则primary不能运行时不会自动切换primary。

总结一下，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，仲裁在数据存储到分片的时候决定存储到哪个节点。

分片 + 副本集 集群搭建

环境准备

系统系统 centos 7
三台服务器：192.168.174.131，192.168.174.132，192.168.174.133
安装包： mongodb-linux-x86_64-3.4.7.tgz

服务器规划

服务器131	服务器132	服务器133
mongos	mongos	mongos
config server	config server	config server
shard server1 主节点	shard server1 从节点	shard server1 仲裁节点
shard server2 仲裁节点	shard server2 主节点	shard server2 从节点
shard server3 从节点	shard server3 仲裁节点	shard server3 主节点

端口分配：

mongos：20000
config：21000
shard1：27017
shard2：27018
shard3：27019

分别在每台机器建立conf、mongos、config、shard1、shard2、shard3六个目录，因为mongos不存储数据，只需要建立日志文件目录即可。

```
mkdir -p /usr/local/mongodb/conf
mkdir -p /usr/local/mongodb/mongos/log
mkdir -p /usr/local/mongodb/config/data
mkdir -p /usr/local/mongodb/config/log
mkdir -p /usr/local/mongodb/shard1/data
mkdir -p /usr/local/mongodb/shard1/log
mkdir -p /usr/local/mongodb/shard2/data
mkdir -p /usr/local/mongodb/shard2/log
mkdir -p /usr/local/mongodb/shard3/data
mkdir -p /usr/local/mongodb/shard3/log
```

config server配置服务器

添加配置文件

```
vi  /usr/local/mongodb/conf/config.conf
##config file content
pidfilepath = /usr/local/mongodb/config/log/configsrv.pid
dbpath = /usr/local/mongodb/config/data
logpath = /usr/local/mongodb/config/log/congigsrv.log
logappend = true

bind_ip = 0.0.0.0
port = 21000
fork = true

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

#replSet name
replSet=configs

# max conn count
maxConns=20000
```

分别启动三台服务器的config server

```
mongod -f /usr/local/mongodb/conf/config.conf
```

登录任意一台配置服务器，初始化配置副本集

```
 #conn
mongo --port 21000
#config value
config = {
      _id : "configs",
members : [
  {_id : 0, host : "192.168.174.131:21000" },
  {_id : 1, host : "192.168.174.132:21000" },
  {_id : 2, host : "192.168.174.133:21000" }
 ]
}

#init  replSet
rs.initiate(config)
```

其中，”_id” : “configs”应与配置文件中配置的 replicaction.replSetName 一致，”members” 中的 “host” 为三个节点的 ip 和 port

配置分片副本集(三台机器)

设置第一个分片副本集

配置文件

vi /usr/local/mongodb/conf/shard1.conf

```
#content
pidfilepath = /usr/local/mongodb/shard1/log/shard1.pid
dbpath = /usr/local/mongodb/shard1/data
logpath = /usr/local/mongodb/shard1/log/shard1.log
logappend = true

bind_ip = 0.0.0.0
port = 27017
fork = true

httpinterface=true
rest=true

#shard name
replSet=shard1

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

#max conns
maxConns=20000
```

启动三台服务器的shard1 server

```
mongod -f /usr/local/mongodb/conf/shard1.conf
```

登陆任意一台服务器，初始化副本集

```
mongo --port 27017
#使用admin数据库
use admin
#定义副本集配置，第三个节点的 "arbiterOnly":true 代表其为仲裁节点。
config = {
_id : "shard1",
members : [
{_id : 0, host : "192.168.174.131:27017" },
{_id : 1, host : "192.168.174.132:27017" },
{_id : 2, host : "192.168.174.133:27017" , arbiterOnly: true }]}
#初始化副本集配置
rs.initiate(config);
```

设置第二个分片副本集

配置文件

vi /usr/local/mongodb/conf/shard2.conf

```
#配置文件内容
# content
```

```
pidfilepath = /usr/local/mongodb/shard2/log/shard2.pid
dbpath = /usr/local/mongodb/shard2/data
logpath = /usr/local/mongodb/shard2/log/shard2.log
logappend = true

bind_ip = 0.0.0.0
port = 27018
fork = true

#open web monitor
httpinterface=true
rest=true

#replSet name
replSet=shard2

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

#max conns
maxConns=20000
```

启动三台服务器的shard2 server

```
./mongod -f /usr/local/mongodb/conf/shard2.conf
```

登陆任意一台服务器，初始化副本集

```
mongo --port 27018
#使用admin数据库
use admin
#定义副本集配置
config = {
_id : "shard2",
  members : [
    {_id : 0, host : "192.168.174.131:27018" , arbiterOnly: true },
    {_id : 1, host : "192.168.174.132:27018" },
    {_id : 2, host : "192.168.174.133:27018" }
  ]
}

#初始化副本集配置
rs.initiate(config);
```

设置第三个分片副本集

配置文件

```
vi /usr/local/mongodb/conf/shard3.conf

#配置文件内容
#content
pidfilepath = /usr/local/mongodb/shard3/log/shard3.pid
dbpath = /usr/local/mongodb/shard3/data
```

```
logpath = /usr/local/mongodb/shard3/log/shard3.log
logappend = true

bind_ip = 0.0.0.0
port = 27019
fork = true

#open web monitor
httpinterface=true
rest=true

#replSet name
replSet=shard3

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

#set max conns
maxConns=20000
```

启动三台服务器的shard3 server

```
./mongod -f /usr/local/mongodb/conf/shard3.conf
```

登陆任意一台服务器，初始化副本集

```
./mongo --port 27019
#使用admin数据库
use admin
#定义副本集配置
config = {
  _id : "shard3",
 members : [
    {_id : 0, host : "192.168.174.131:27019" },
    {_id : 1, host : "192.168.174.132:27019" , arbiterOnly: true},
   {_id : 2, host : "192.168.174.133:27019" }
 ]
 }

#初始化副本集配置
rs.initiate(config);
```

配置路由服务器 mongos

先启动配置服务器和分片服务器,后启动路由实例启动路由实例:（三台机器）

```
vi /usr/local/mongodb/conf/mongos.conf

#content
pidfilepath = /usr/local/mongodb/mongos/log/mongos.pid
logpath = /usr/local/mongodb/mongos/log/mongos.log
logappend = true
```

bind_ip = 0.0.0.0

port = 20000

fork = true

#monitor setting server, only 1 or 3, config is config server respSet name

configdb = configs/192.168.174.131:21000,192.168.174.132:21000,192.168.174.133:21000

#set max conns

maxConns=20000

启动三台服务器的mongos server

./mongos -f /usr/local/mongodb/conf/mongos.conf

启用分片

目前搭建了mongodb配置服务器、路由服务器，各个分片服务器，不过应用程序连接到mongos路由服务器并不能使用分片机制，还需要在程序里设置分片配置，让分片生效。

登陆任意一台mongos

mongo --port 20000

#使用admin数据库

use admin

#串联路由服务器与分配副本集

sh.addShard("shard1/192.168.174.131:27017,192.168.174.132:27017,192.168.174.133:27017")

sh.addShard("shard2/192.168.174.131:27018,192.168.174.132:27018,192.168.174.133:27018")

sh.addShard("shard3/192.168.174.131:27019,192.168.174.132:27019,192.168.174.133:27019")

#查看集群状态

sh.status()

```
mongos> show dbs;
admin   0.000GB
config  0.000GB
mongos> use admin;
switched to db admin
mongos>
mongos> sh.addShard("shard1/192.168.174.131:27017,192.168.174.132:27017,192.168.174.133:27017")
{ "shardAdded" : "shard1", "ok" : 1 }
mongos> sh.addShard("shard1/192.168.174.131:27018,192.168.174.132:27018,192.168.174.133:27018")
{ "shardAdded" : "shard1", "ok" : 1 }
mongos> sh.addShard("shard2/192.168.174.131:27018,192.168.174.132:27018,192.168.174.133:27018")
{ "shardAdded" : "shard2", "ok" : 1 }
mongos> sh.addShard("shard3/192.168.174.131:27019,192.168.174.132:27019,192.168.174.133:27019")
{ "shardAdded" : "shard3", "ok" : 1 }
mongos> sh.status();
--- Sharding Status ---
  sharding version: {
        "_id" : 1,
        "minCompatibleVersion" : 5,
        "currentVersion" : 6,
        "clusterId" : ObjectId("59acdce64273c5b70145cbce")
}
  shards:
        {  "_id" : "shard1",  "host" : "shard1/192.168.174.131:27017,192.168.174.132:27017",  "state" : 1 }
        {  "_id" : "shard2",  "host" : "shard2/192.168.174.132:27018,192.168.174.133:27018",  "state" : 1 }
        {  "_id" : "shard3",  "host" : "shard3/192.168.174.131:27019,192.168.174.133:27019",  "state" : 1 }
  active mongoses:
        "3.4.7" : 1
 autosplit:
        Currently enabled: yes
  balancer:
        Currently enabled:  yes
        Currently running:  no
                Balancer lock taken at Mon Sep 04 2017 12:56:06 GMT+0800 (CST) by ConfigServer:Balancer
        Failed balancer rounds in last 5 attempts:  0
        Migration Results for the last 24 hours:
                No recent migrations
  databases:
```

测试

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。

#指定order数据库分片生效
db.runCommand({ enablesharding :"river"});
#指定数据库里需要分片的集合和片键
db.runCommand({ shardcollection : "river.order",key : {id: 1} })
我们设置river的 order表需要分片，根据 id 自动分片到 shard1 ，shard2，shard3 上面去。要这样设置是因为不是所有 mongodb 的数据库和表 都需要分片！
测试分片配置结果
mongo 127.0.0.1:20000
#使用river
use river;
#插入测试数据
for (var i = 1; i <= 100000; i++){
db.order.insert({id:i,"name":"order"+i})};
#查看分片情况如下，部分无关信息省掉了
db.order.stats();

可以看到数据分到3个分片，各自分片数量为： shard1 “count” : 0，shard2 “count” : 20，shard3 “count” : 99980。已经成功了！

后期运维

启动关闭
mongodb的启动顺序是，先启动配置服务器，在启动分片，最后启动mongos.

```
mongod -f /usr/local/mongodb/conf/config.conf
mongod -f /usr/local/mongodb/conf/shard1.conf
mongod -f /usr/local/mongodb/conf/shard2.conf
mongod -f /usr/local/mongodb/conf/shard3.conf
mongod -f /usr/local/mongodb/conf/mongos.conf
```

关闭时，直接killall杀掉所有进程(一般不需要关闭所有集群)

```
killall mongod
killall mongos
```

2017年6月全球数据库排行榜TOP20

微信公众号：商业排行榜　　官方网站：http://top.askci.com

排名	数据库管理系统	数据库模型	得分
1	Oracle	Relational DBMS	1351.76
2	MySQL	Relational DBMS	1345.31
3	Microsoft SQL Server	Relational DBMS	1198.97
4	PostgreSQL	Relational DBMS	368.54
5	MongoDB	Document store	335.00
6	DB2	Relational DBMS	187.50
7	Microsoft Access	Relational DBMS	126.55
8	Cassandra	Wide column store	124.12
9	Redis	Key-value store	118.89
10	SQLite	Relational DBMS	116.71
11	Elasticsearch	Search engine	111.56
12	Teradata	Relational DBMS	77.33
13	SAP Adaptive Server	Relational DBMS	67.52
14	Solr	Search engine	63.61
15	HBase	Wide column store	61.87
16	Splunk	Search engine	57.52
17	FileMaker	Relational DBMS	57.08
18	MariaDB	Relational DBMS	52.89
19	SAP HANA	Relational DBMS	47.50
20	Hive	Relational DBMS	44.38