# 分片

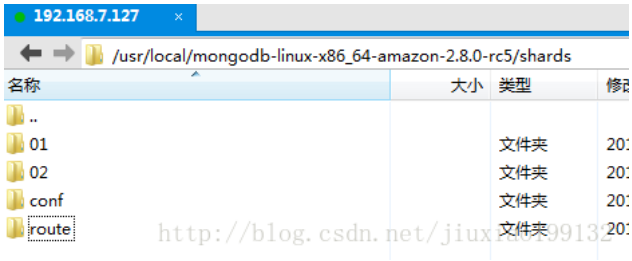
在Mongodb里面存在另一种集群，就是分片技术,可以满足MongoDB数据量大量增长的需求。
当MongoDB存储海量的数据时，一台机器可能不足以存储数据，也可能不足以提供可接受的读写吞吐量。这时，我们就可以通过在多台机器上分割数据，使得数据库系统能存储和处理更多的数据

# 配置分片单节点

## 准备工作 注意这里的版本不是3.4 mongodb安装过程不再重复


需要两个mongd实例 对应 分片 A 与 分片B ， 端口 如图， data存放路径如图
需要一个配置服务器 同上 端口 data路径
路由同上

新建目录结构为




## 启动分片A

```
cd /usr/local/mongodb-linux-x86_64-amazon-2.8.0-rc5/bin/
./mongod --shardsvr --port 30001 --dbpath /usr/local/mongodb-linux-x86_64-amazon-2.8.0-rc5/shards/01
```

执行成功，输出结果，进程ID2745

```
about to fork child process, waiting until server is ready for connections.
forked process: 2745
child process started successfully, parent exiting
```

## 启动分片B

先建立好文件夹

```
/usr/local/mongodb-linux-x86_64-amazon-2.8.0-rc5/shards/02
```

输入命令

```
./mongod --shardsvr --port 30002 --dbpath /usr/local/mongodb-linux-x86_64-amazon-2.8.0-rc5/shards/02
```

输出

```
about to fork child process, waiting until server is ready for connections.
forked process: 2757
child process started successfully, parent exiting
```

## 启动配置服务

创建好文件夹

```
/usr/local/mongodb-linux-x86_64-amazon-2.8.0-rc5/shards/conf
```

输入命令

```
./mongod --port 30000 --configsvr --dbpath /usr/local/mongodb-linux-x86_64-amazon-2.8.0-rc5/shards/c
```

输出结果

```
about to fork child process, waiting until server is ready for connections.
forked process: 2782
child process started successfully, parent exiting
```

## 启动路由

新建文件夹

```
/usr/local/mongodb-linux-x86_64-amazon-2.8.0-rc5/shards/route
```

输入命令

```
./mongos --port 40000 --configdb 192.168.7.127:30000 --logpath /usr/local/mongodb-linux-x86_64-amazo
```

输出结果
这里注意输出中提示，分片中只有一个单节点建议用于测试，不建议用于生产环境，后面会配置副本集的形式

```
2017-05-09T14:04:44.601+0800 W SHARDING running with 1 config server should be done only for testing
about to fork child process, waiting until server is ready for connections.
forked process: 2795
child process started successfully, parent exiting
```

## 配置分片集群

登录到路由(mongos)这个节点（这里用的Xshell直接连接，非MongoDB客户端）

```
[root@localhost bin]# ./mongo --port 40000
```

添加分片A

```
mongos> sh.addShard("192.168.7.127:30001");
```

输出

```
{ "shardAdded" : "shard0000", "ok" : 1 }
```

添加分片B

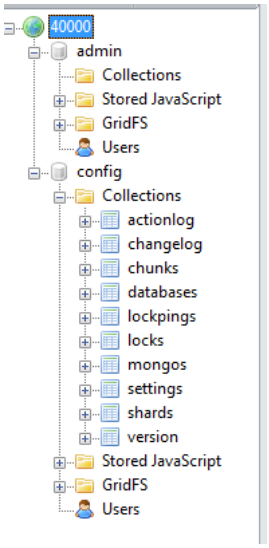
```
mongos> sh.addShard("192.168.7.127:30002");
```

输出

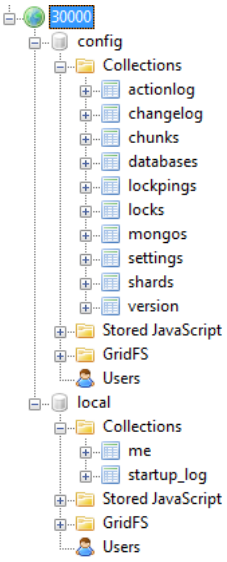
```
{ "shardAdded" : "shard0001", "ok" : 1 }
```

## 查看不同节点的数据库和表

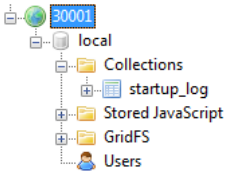
### 路由节点



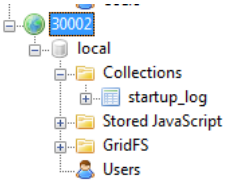

### 配置节点




## 分片A




## 分片B




可以看到 路由节点 和 配置节点 都存在 一个config数据库，这个数据库存放了分片的信息

# 查看分片

查看分片

```
mongos> db.getSiblingDB("config").shards.find();
```

输出

```
{ "_id" : "shard0000", "host" : "192.168.7.127:30001" }
{ "_id" : "shard0001", "host" : "192.168.7.127:30002" }
```

查看分片对哪些数据库有效

```
mongos> db.getSiblingDB("config").databases.find()
```

输出

```
{ "_id" : "admin", "partitioned" : false, "primary" : "config" }
{ "_id" : "test", "partitioned" : false, "primary" : "shard0000" }
```

开启一个数据库的分片

```
mongos> sh.enableSharding("cloud-docs");
```

输出

```
{ "ok" : 1 }
```

再来查看哪些数据库开启了分片

```
mongos> db.getSiblingDB("config").databases.find()
```

可以看到多出一个cloud-docs 数据库

```
{ "_id" : "admin", "partitioned" : false, "primary" : "config" }
{ "_id" : "test", "partitioned" : false, "primary" : "shard0000" }
{ "_id" : "cloud-docs", "partitioned" : true, "primary" : "shard0000" }
```

对数据库的book表 进行分片， 分片键 这里使用 year 和 id 的组合分片键

```
mongos> sh.shardCollection("cloud-docs.book",{"year":1,"_id":1});
```

输出

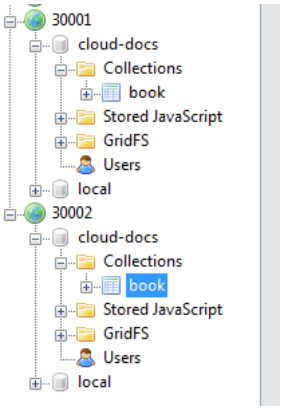
```
{ "collectionsharded" : "cloud-docs.book", "ok" : 1 }
```

## 分片写入测试写入一千条数据

链接路由节点
指定刚才设定启用分片的数据库cloud-docs，和指定的表book

```
@Test
public void initDate(){
    MongoDatabase db = new MongoClient( "192.168.7.127", 40000).getDatabase("cloud-docs");
    MongoCollection<Document> coll = db.getCollection("book");
    for(int i = 0 ; i < 1000 ; i ++){
        Document doc = new Document();
        User u = User.initUser();
        doc.put("address", u.getAddress());
        doc.put("year", u.getAge());
         doc.put("email", u.getEmail());
         doc.put("height", u.getHeight());
         doc.put("job", u.getJob());
         doc.put("nickname", u.getNickname());
         doc.put("phone", u.getPhone());
         doc.put("school", u.getSchool());
         doc.put("sex", u.getSex());
         doc.put("hoby", u.getHoby());
         Document dog = new Document();
         dog.put("name", u.getDog().getName());
         dog.put("age", u.getDog().getAge());
         doc.put("dog", dog);
         coll.insertOne(doc);
     }
}

```




查看 分片A 和 分别B 两个数据库 book 加起来 值刚好等于 1000