Chapter 3: Sharding

Lab - Configure a Sharded Cluster

1. Bring up the config server replica set (CSRS)

Here are the remaining two config files for `m103-csrs`:

CSRS Node 2:

COPY

```
sharding:
  clusterRole: configsvr
replication:
  replSetName: m103-csrs
security:
  keyFile: /var/mongodb/pki/m103-keyfile
net:
  bindIp: localhost,192.168.103.100
  port: 26002
systemLog:
  destination: file
  path: /var/mongodb/db/csrs2.log
  logAppend: true
processManagement:
  fork: true
storage:
  dbPath: /var/mongodb/db/csrs2
```

CSRS Node 3:

COPY

```
sharding:
  clusterRole: configsvr
replication:
  replSetName: m103-csrs
security:
  keyFile: /var/mongodb/pki/m103-keyfile
net:
  bindIp: localhost,192.168.103.100
  port: 26003
systemLog:
  destination: file
  path: /var/mongodb/db/csrs3.log
  logAppend: true
processManagement:
  fork: true
storage:
  dbPath: /var/mongodb/db/csrs3
```

Once all three mongod processes are running, we can bring up `m103-csrs` just like any other replica set.

2. Bring up the mongos

Once we have the config file for our mongos process (`mongos.conf`), we can navigate to the directory where it is located and bring up the mongos:

```
mongos -f mongos.conf
```

No further configuration of mongos is required for this lab.

3. Reconfigure `m103-repl`

Here are the updated config files for the three nodes in `m103-repl`:

Node 1:

```
storage:
  dbPath: /var/mongodb/db/1
  wiredTiger:
     engineConfig:
        cacheSizeGB: .1
net:
  bindIp: 192.168.103.100,localhost
  port: 27001
security:
  keyFile: /var/mongodb/pki/m103-keyfile
systemLog:
  destination: file
  path: /var/mongodb/db/mongod1/mongod.log
  logAppend: true
processManagement:
  fork: true
operationProfiling:
  slowOpThresholdMs: 50
replication:
  replSetName: m103-repl
sharding:
  clusterRole: shardsvr
```

Node 2:

```
storage:
  dbPath: /var/mongodb/db/2
  wiredTiger:
     engineConfig:
        cacheSizeGB: .1
net:
  bindIp: 192.168.103.100,localhost
  port: 27002
security:
  keyFile: /var/mongodb/pki/m103-keyfile
systemLog:
  destination: file
  path: /var/mongodb/db/mongod2/mongod.log
  logAppend: true
processManagement:
  fork: true
operationProfiling:
  slowOpThresholdMs: 50
replication:
  replSetName: m103-repl
sharding:
  clusterRole: shardsvr
```

Node 3:

COPY

```
storage:
  dbPath: /var/mongodb/db/3
  wiredTiger:
     engineConfig:
        cacheSizeGB: .1
net:
  bindIp: 192.168.103.100,localhost
  port: 27003
security:
  keyFile: /var/mongodb/pki/m103-keyfile
systemLog:
  destination: file
  path: /var/mongodb/db/mongod3/mongod.log
  logAppend: true
processManagement:
  fork: true
operationProfiling:
  slowOpThresholdMs: 50
replication:
  replSetName: m103-repl
sharding:
  clusterRole: shardsvr
```

Once all three mongod processes are restarted, `m103-repl` is enabled for sharding.

4. Add `m103-repl` as the first shard

From the mongo shell of our mongos, we can run the following command to add our replica set as a shard in this cluster:

```
sh.addShard("m103-repl/192.168.103.100:27001")
```

COPY

Note that we only need to specify one node in the replica set of our new shard. The response of this command should contain something like this:

```
{
  "shardAdded" : "m103-repl"
}
```

Running `sh.status()` should show our new shard in the cluster:

```
shards:
        {  "_id" : "m103-repl",  "host" : "m103-repl/192.168.103.100:27001,192.168.103.100:27002,192.168.103.100:27003",  "state" : 1 }
```

From this point onward, all CRUD operations will go through mongos. We need not connect to individual shards, as mongos will route our queries for us.

Proceed to next section