FINAL: QUESTION 9

Now that the config server from question #8 is up and running, we will restore the two shards ("s1" and "s2").

If we inspect our restored config db, we see this in db.shards:

```
~/dba/final $ mongo localhost:27019/config
MongoDB shell version: 3.0.0
connecting to: localhost:27019/config
configsvr> db.shards.find()
{ "_id" : "s1", "host" : "s1/genome_svr1:27501,genome_svr2:27502,genome_svr2:2
7503" }
{ "_id" : "s2", "host" : "s2/genome_svr4:27601,genome_svr5:27602,genome_svr5:2
7603" }
```

From this we know when we run a `mongos` for the cluster, it will expect the first shard to be a replica set named "s1", and the second to be a replica set named "s2", and also to be able to be able to resolve and connect to at least one of the seed hostnames for each shard.

If we were restoring this cluster as "itself", it would be best to assign the hostnames "genome_svr1" etc. to the appropriate IP addresses in DNS, and not change `config.shard`. However, for this problem, our job is not to restore the cluster, but rather to create a new temporary data mart initialized with this dataset.

Thus instead we will update the config.shards metadata to point to the locations of our new shard servers. Update the config.shards collection such that your output is:

```
configsvr> db.shards.find()
{ "_id" : "s1", "host" : "localhost:27501" }
{ "_id" : "s2", "host" : "localhost:27601" }
configsvr>
```

Be sure when you do this nothing is running except the single config server. mongod and mongos processes cache metadata, so this is important. After the update restart the config server itself for the same reason.

Now start a mongod for each shard -- one on port 27501 for shard "s1" and on port 27601 for shard "s2". At this point if you run ps you should see three mongod's -- one for each shard, and one for our config server. Note they need not be replica sets, but just regular mongod's, as we did not begin our `host` string in config.shards with `setname/`. Finally, use `mongorestore` to restore the data for each shard.

The next step is to start a `mongos` for the cluster.

Connect to the mongos with a `mongo` shell. Run this:

```
> use snps
> var x = db.elegans.aggregate( [ { $match : { N2 : "T" } } , { $group : { _id
  :"$N2" , n : { $sum : 1 } } } ] ).next(); print( x.n )
```

Enter the number output for `n`.

Notes:

- *You must do this with MongoDB 3.0. The mongoimport may not work with prior versions of MongoDB.*

1

SUBMIT