MSC DSAI L010
Rehmah Ahmed Batki

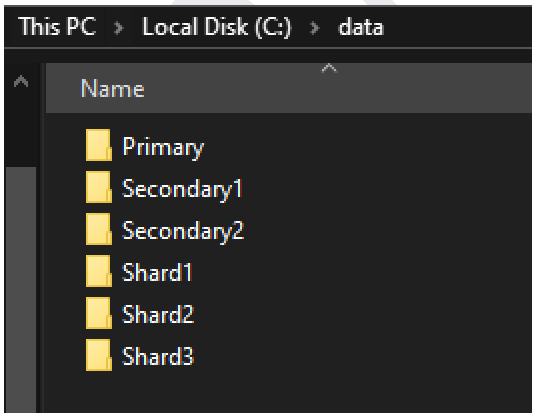
Practical 7
21-2-25
Sharding using Mongodb

Write-up: -
- Sharding
- Sharding Architecture
- Sharding Types, benefits and limitations

You are a database administrator tasked with setting up MongoDB sharding to handle a growing dataset. You need to configure a MongoDB sharded cluster with the following requirements:
- One Config Server (Replica set)
- Two Shard Servers
- One Mongos Router
- Sharding enabled for a database and collection

https://gist.github.com/DGamer007/0864c6aeebf27e3821602d9dd5ca7375

Create the following folders


This PC › Local Disk (C:) › data

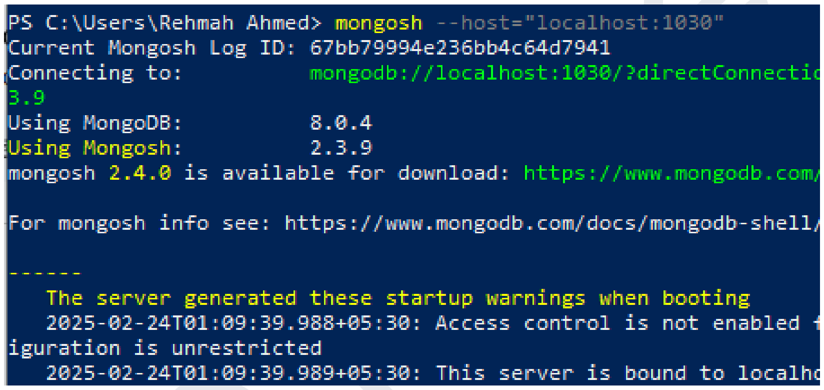
Name
Primary
Secondary1
Secondary2
Shard1
Shard2
Shard3

Creating 3 Replica Servers
In 3 powershells:

```
mongod --configsvr --port=1030 --replSet="Prac7Repl"
--dbpath="E:\data\Primary"

mongod --configsvr --port=1040 --replSet="Prac7Repl"
--dbpath="E:\data\Secondary1"

mongod --configsvr --port=1050 --replSet="Prac7Repl"
--dbpath="E:\data\Secondary2"
```

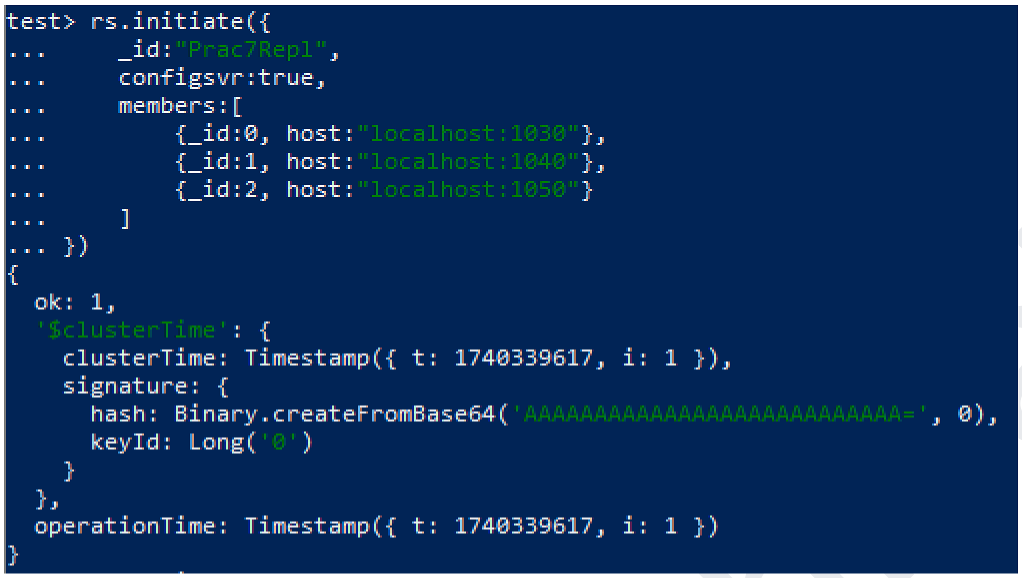
Connect to anyone of them using mongosh & Initiate Replica Set
In a new powershell

```
mongosh --host="localhost:1030"
```



```
PS C:\Users\Rehmah Ahmed> mongosh --host="localhost:1030"
Current Mongosh Log ID: 67bb79994e236bb4c64d7941
Connecting to:          mongodb://localhost:1030/?directConnecti
3.9
Using MongoDB:          8.0.4
Using Mongosh:          2.3.9
mongosh 2.4.0 is available for download: https://www.mongodb.com/

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

------
   The server generated these startup warnings when booting
   2025-02-24T01:09:39.988+05:30: Access control is not enabled f
iguration is unrestricted
   2025-02-24T01:09:39.989+05:30: This server is bound to localho
```

```
rs.initiate({
   _id:"Prac7Repl",
   configsvr:true,
   members:[
      {_id:0, host:"localhost:1030"},
      {_id:1, host:"localhost:1040"},
      {_id:2, host:"localhost:1050"}
   ]
})
```



```
test> rs.initiate({
...     _id:"Prac7Repl",
...     configsvr:true,
...     members:[
...         {_id:0, host:"localhost:1030"},
...         {_id:1, host:"localhost:1040"},
...         {_id:2, host:"localhost:1050"}
...     ]
... })
{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1740339617, i: 1 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1740339617, i: 1 })
}
```

Creating 3 Shard Instances
In 3 powershells:

mongod --shardsvr --port=1130 --dbpath="E:\data\Shard1" --replSet="Prac7Shard"

mongod --shardsvr --port=1140 --dbpath="E:\data\Shard2" --replSet="Prac7Shard"

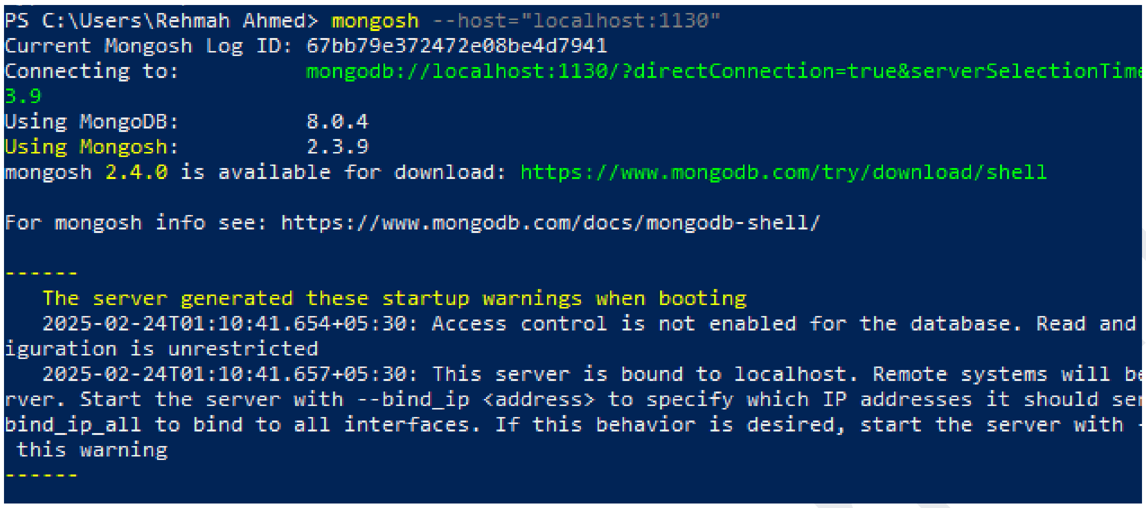
mongod --shardsvr --port=1150 --dbpath="E:\data\Shard3" --replSet="Prac7Shard"

Connect to anyone of them using mongosh and Initiate Shard
In a new powershell

mongosh --host="localhost:1130"

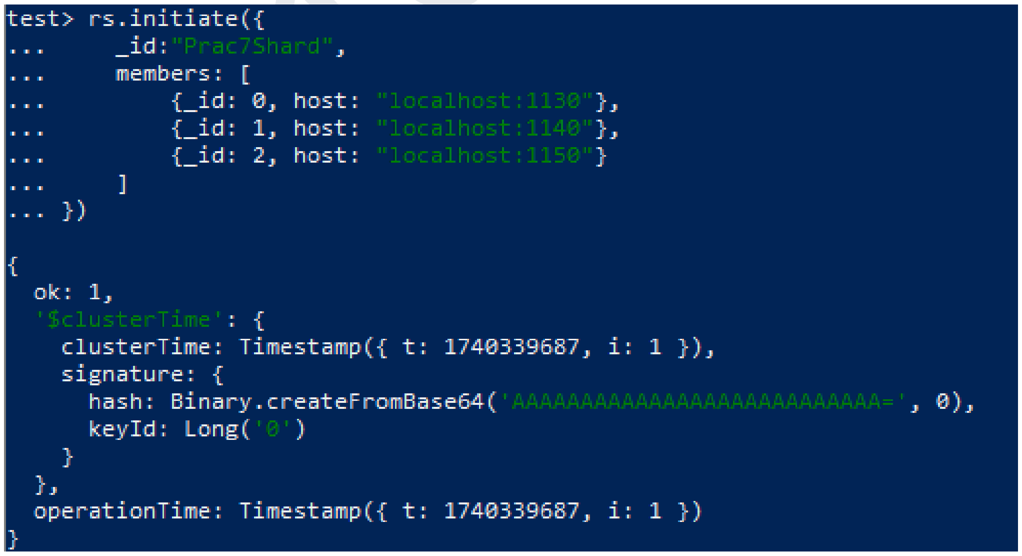
MSC DSAI L010

Rehmah Ahmed Batki



```
PS C:\Users\Rehmah Ahmed> mongosh --host="localhost:1130"
Current Mongosh Log ID: 67bb79e372472e08be4d7941
Connecting to:          mongodb://localhost:1130/?directConnection=true&serverSelectionTime
3.9
Using MongoDB:          8.0.4
Using Mongosh:          2.3.9
mongosh 2.4.0 is available for download: https://www.mongodb.com/try/download/shell

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

------
   The server generated these startup warnings when booting
   2025-02-24T01:10:41.654+05:30: Access control is not enabled for the database. Read and
iguration is unrestricted
   2025-02-24T01:10:41.657+05:30: This server is bound to localhost. Remote systems will be
rver. Start the server with --bind_ip <address> to specify which IP addresses it should se
bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with -
 this warning
------
```

```
rs.initiate({
   _id:"Prac7Shard",
   members: [
      {_id: 0, host: "localhost:1130"},
      {_id: 1, host: "localhost:1140"},
      {_id: 2, host: "localhost:1150"}
   ]
})
```



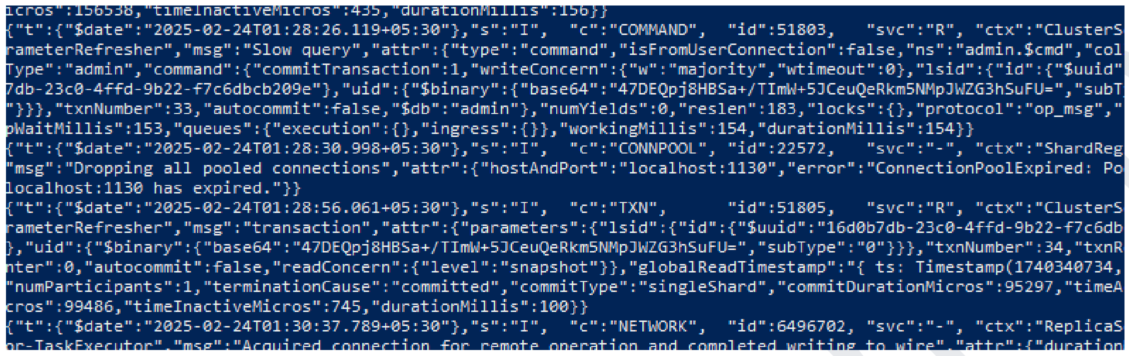
```
test> rs.initiate({
...      _id:"Prac7Shard",
...      members: [
...          {_id: 0, host: "localhost:1130"},
...          {_id: 1, host: "localhost:1140"},
...          {_id: 2, host: "localhost:1150"}
...      ]
... })

{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1740339687, i: 1 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1740339687, i: 1 })
}
```

MSC DSAI L010

Rehmah Ahmed Batki 🎀5🎀

Initialize a Query Router which is a mongos process.

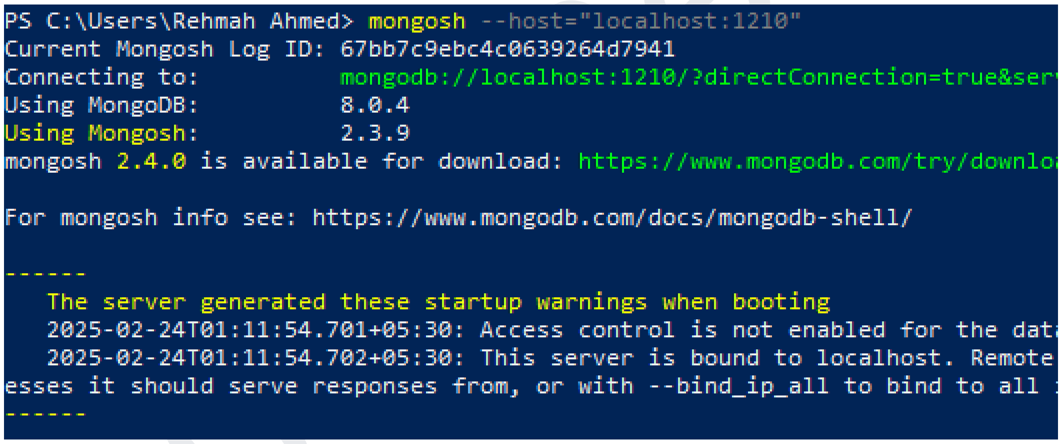
mongos --port=1210 --configdb="Prac7Repl/localhost:1030,localhost:1040,localhost:1050"



```
icros":156538,"timeInactiveMicros":435,"durationMillis":156}}
{"t":{"$date":"2025-02-24T01:28:26.119+05:30"},"s":"I",  "c":"COMMAND",  "id":51803,   "svc":"R", "ctx":"ClusterS
rameterRefresher","msg":"Slow query","attr":{"type":"command","isFromUserConnection":false,"ns":"admin.$cmd","col
Type":"admin","command":{"commitTransaction":1,"writeConcern":{"w":"majority","wtimeout":0},"lsid":{"id":{"$uuid"
7db-23c0-4ffd-9b22-f7c6dbcb209e"},"uid":{"$binary":{"base64":"47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=","subT
"}}},"txnNumber":33,"autocommit":false,"$db":"admin"},"numYields":0,"reslen":183,"locks":{},"protocol":"op_msg","
pWaitMillis":153,"queues":{"execution":{},"ingress":{}},"workingMillis":154,"durationMillis":154}}
{"t":{"$date":"2025-02-24T01:28:30.998+05:30"},"s":"I",  "c":"CONNPOOL", "id":22572,   "svc":"-", "ctx":"ShardReg
"msg":"Dropping all pooled connections","attr":{"hostAndPort":"localhost:1130","error":"ConnectionPoolExpired: Po
localhost:1130 has expired."}}
{"t":{"$date":"2025-02-24T01:28:56.061+05:30"},"s":"I",  "c":"TXN",      "id":51805,   "svc":"R", "ctx":"ClusterS
rameterRefresher","msg":"transaction","attr":{"parameters":{"lsid":{"id":{"$uuid":"16d0b7db-23c0-4ffd-9b22-f7c6db
},"uid":{"$binary":{"base64":"47DEQpj8HBSa+/TImW+5JCeuQeRkm5NMpJWZG3hSuFU=","subType":"0"}}},"txnNumber":34,"txnR
nter":0,"autocommit":false,"readConcern":{"level":"snapshot"}},"globalReadTimestamp":"{ ts: Timestamp(1740340734,
"numParticipants":1,"terminationCause":"committed","commitType":"singleShard","commitDurationMicros":95297,"timeA
cros":99486,"timeInactiveMicros":745,"durationMillis":100}}
{"t":{"$date":"2025-02-24T01:30:37.789+05:30"},"s":"I",  "c":"NETWORK",  "id":6496702, "svc":"-", "ctx":"ReplicaS
or-TaskExecutor","msg":"Acquired connection for remote operation and completed writing to wire","attr":{"duration
```

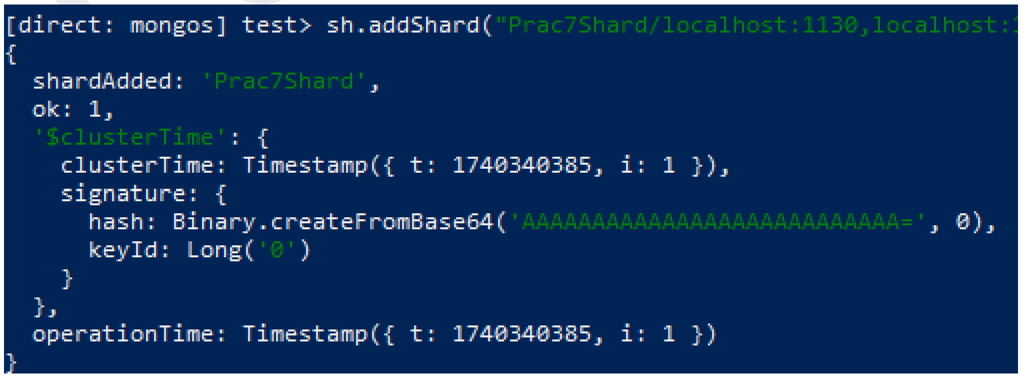
Now, Connect Shards and Query Router (mongos)

mongosh --host="localhost:1210"



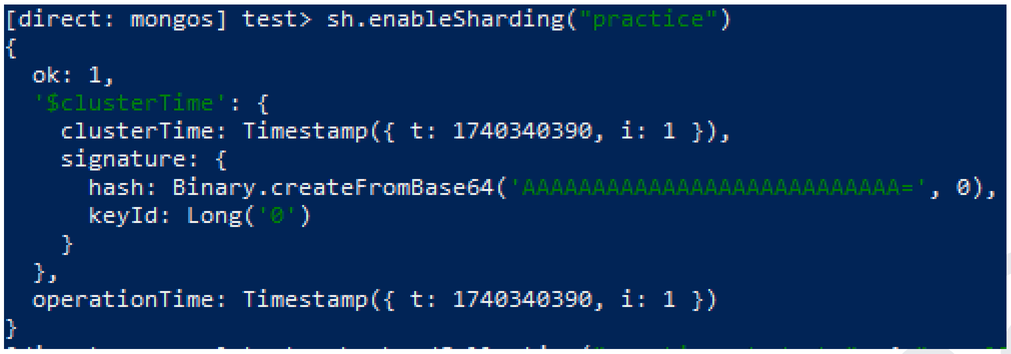
```
PS C:\Users\Rehmah Ahmed> mongosh --host="localhost:1210"
Current Mongosh Log ID: 67bb7c9ebc4c0639264d7941
Connecting to:          mongodb://localhost:1210/?directConnection=true&ser
Using MongoDB:          8.0.4
Using Mongosh:          2.3.9
mongosh 2.4.0 is available for download: https://www.mongodb.com/try/downlo

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

------
   The server generated these startup warnings when booting
   2025-02-24T01:11:54.701+05:30: Access control is not enabled for the dat
   2025-02-24T01:11:54.702+05:30: This server is bound to localhost. Remote
esses it should serve responses from, or with --bind_ip_all to bind to all
------
```

sh.addShard("Prac7Shard/localhost:1130,localhost:1140,localhost:1150")



```
[direct: mongos] test> sh.addShard("Prac7Shard/localhost:1130,localhost:1
{
  shardAdded: 'Prac7Shard',
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1740340385, i: 1 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1740340385, i: 1 })
}
```

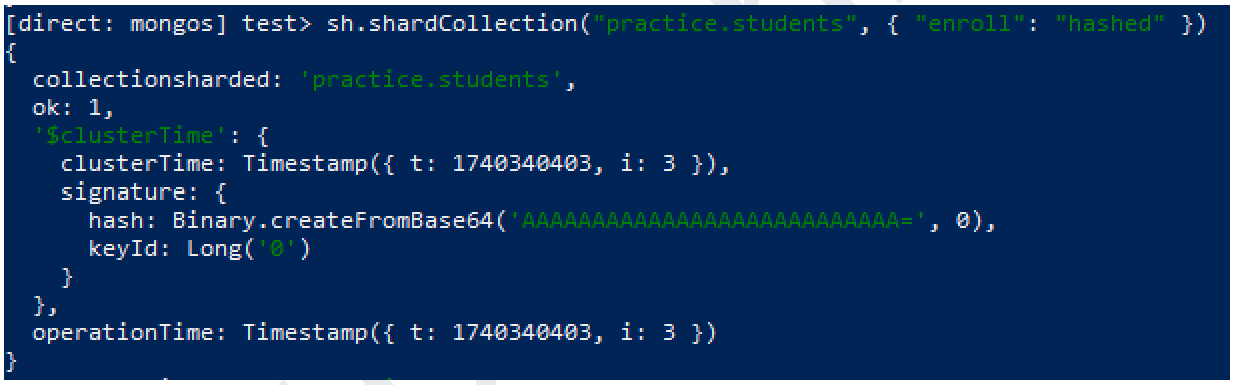
sh.enableSharding("practice")



```
[direct: mongos] test> sh.enableSharding("practice")
{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1740340390, i: 1 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1740340390, i: 1 })
}
```

Shard a Collection on the Sharding Enabled Database

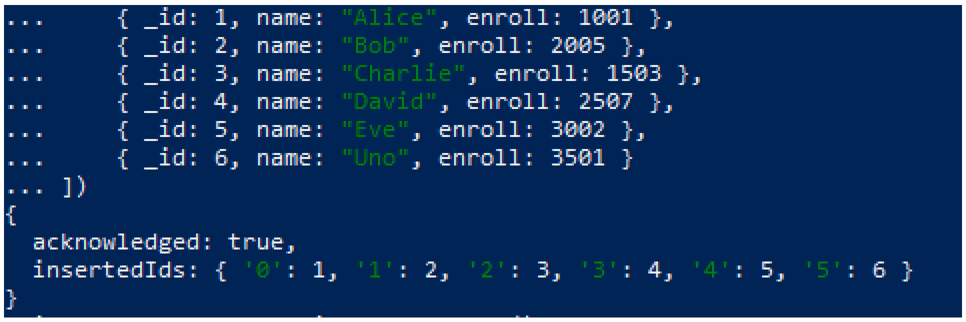
sh.shardCollection("practice.students", { "enroll": "hashed" })



```
[direct: mongos] test> sh.shardCollection("practice.students", { "enroll": "hashed" })
{
  collectionsharded: 'practice.students',
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1740340403, i: 3 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1740340403, i: 3 })
}
```

Inserting 6 values

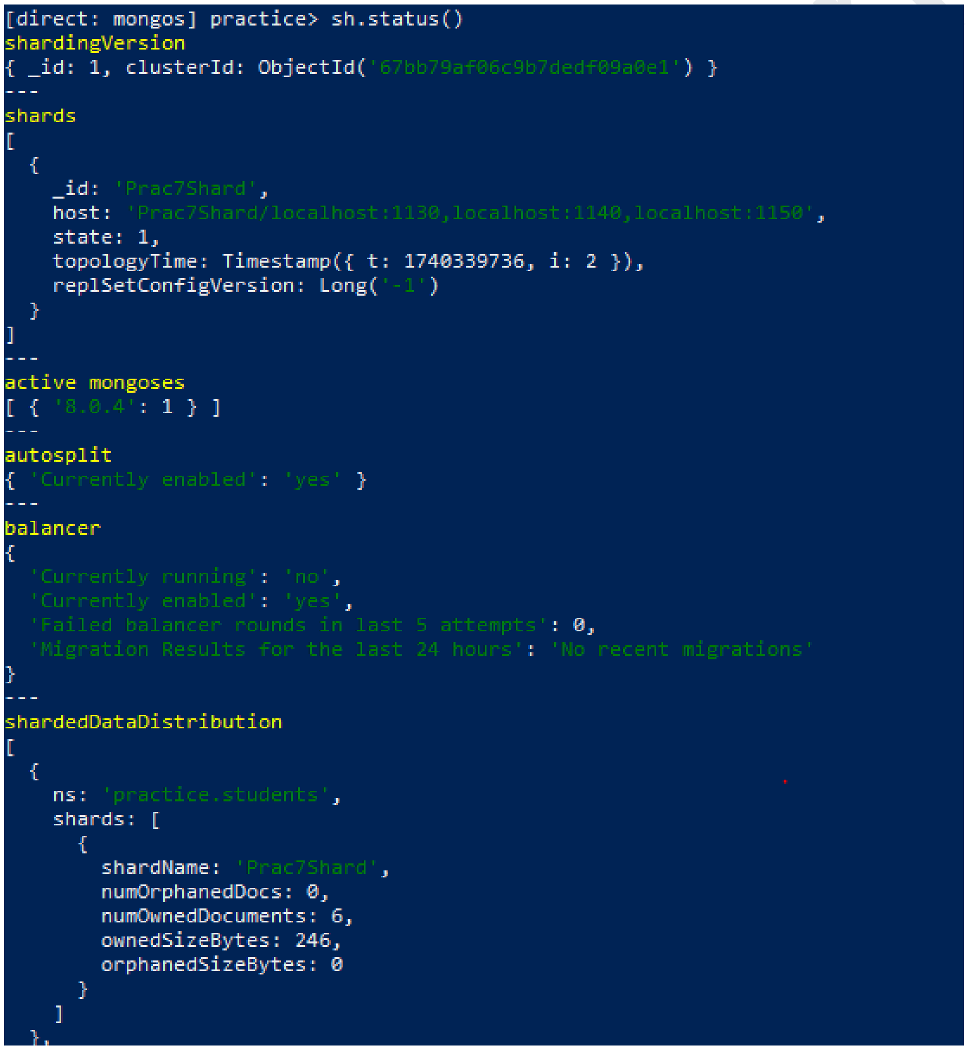
```
use practice
db.students.insertMany([
  { _id: 1, name: "Alice", enroll: 1001 },
  { _id: 2, name: "Bob", enroll: 2005 },
  { _id: 3, name: "Charlie", enroll: 1503 },
  { _id: 4, name: "David", enroll: 2507 },
  { _id: 5, name: "Eve", enroll: 3002 },
  { _id: 6, name: "Uno", enroll: 3501 }
])
```



```
...     { _id: 1, name: "Alice", enroll: 1001 },
...     { _id: 2, name: "Bob", enroll: 2005 },
...     { _id: 3, name: "Charlie", enroll: 1503 },
...     { _id: 4, name: "David", enroll: 2507 },
...     { _id: 5, name: "Eve", enroll: 3002 },
...     { _id: 6, name: "Uno", enroll: 3501 }
... ])
{
  acknowledged: true,
  insertedIds: { '0': 1, '1': 2, '2': 3, '3': 4, '4': 5, '5': 6 }
}
```

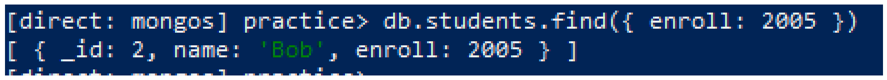
Checking how they are stored

sh.status()



```
[direct: mongos] practice> sh.status()
shardingVersion
{ _id: 1, clusterId: ObjectId('67bb79af06c9b7dedf09a0e1') }
---
shards
[
  {
    _id: 'Prac7Shard',
    host: 'Prac7Shard/localhost:1130,localhost:1140,localhost:1150',
    state: 1,
    topologyTime: Timestamp({ t: 1740339736, i: 2 }),
    replSetConfigVersion: Long('-1')
  }
]
---
active mongoses
[ { '8.0.4': 1 } ]
---
autosplit
{ 'Currently enabled': 'yes' }
---
balancer
{
  'Currently running': 'no',
  'Currently enabled': 'yes',
  'Failed balancer rounds in last 5 attempts': 0,
  'Migration Results for the last 24 hours': 'No recent migrations'
}
---
shardedDataDistribution
[
  {
    ns: 'practice.students',
    shards: [
      {
        shardName: 'Prac7Shard',
        numOrphanedDocs: 0,
        numOwnedDocuments: 6,
        ownedSizeBytes: 246,
        orphanedSizeBytes: 0
      }
    ]
  },
```

Find By Enroll (Efficient)
db.students.find({ enroll: 2005 })



```
[direct: mongos] practice> db.students.find({ enroll: 2005 })
[ { _id: 2, name: 'Bob', enroll: 2005 } ]
```

Range queries (not efficient)
db.students.find({ enroll: { $gt: 1000, $lt: 3000 } })

MongoDB has to check all shards since hashed values are scattered.