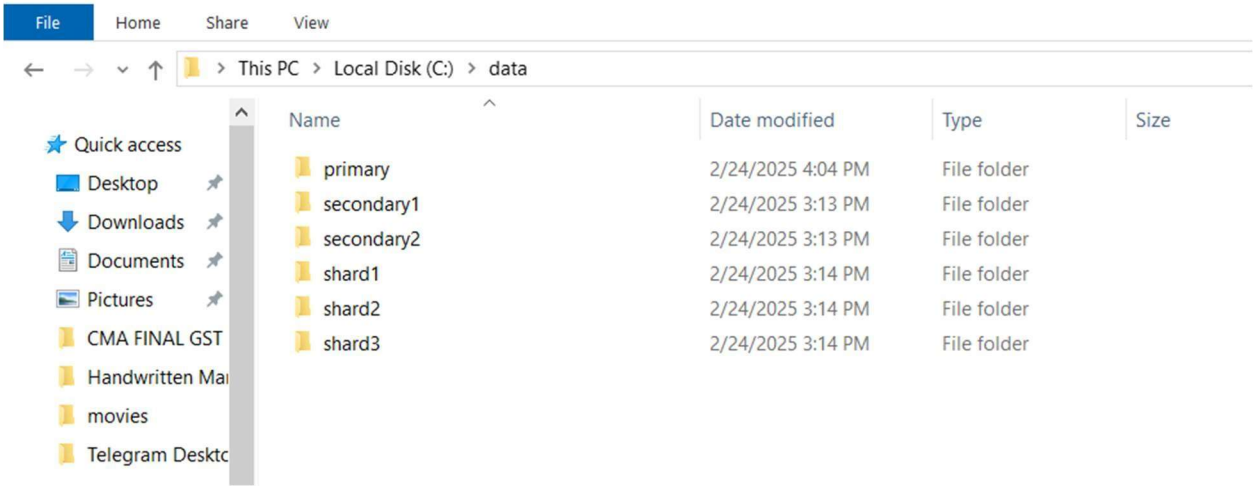

Create 3 replica set

mongod --configsvr --port=1030 --replSet="pract7" --dbpath="C:\data/server1"

mongod --configsvr --port=1040 --replSet="pract7" --dbpath="C:\data/server2"
mongod --configsvr --port=1050 --replSet="pract7" --dbpath="C:\data/server3"

Connect to anyone of them using mongosh and Initiate Replica Set

 # Connect using mongosh mongosh --
host="localhost:1030"

```
PS C:\Users\DELL> mongosh --host="localhost:1030"
Current Mongosh Log ID: 67bc4e717cac0f1bf8fa4213
Connecting to:          mongodb://localhost:1030/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.4.0
Using MongoDB:          7.0.16
Using Mongosh:          2.4.0

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

------
   The server generated these startup warnings when booting
   2025-02-24T16:15:24.730+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
   2025-02-24T16:15:24.731+05:30: This server is bound to localhost. Remote systems will be unable to connect to this server. Start the server with
 --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable this warning
------
```

 # Initiate ReplicaSet with Configurations rs.initiate({
_id:"pract7", configsvr:true, members:[
{_id:0, host:"localhost:1030"},
{_id:1, host:"localhost:1040"},
{_id:2, host:"localhost:1050"}
]
})

```
test> rs.initiate({
...     _id:"pract7",
...     configsvr:true,
...     members:[
...         {_id:0, host:"localhost:1030"},
...         {_id:1, host:"localhost:1040"},
...         {_id:2, host:"localhost:1050"}
...     ]
... })
{ ok: 1 }
pract7 [direct: other] test>
```

 # Initialize MongoDB Shards

mongod --shardsvr --port=1130 --dbpath=" C:\data/shard1" -replSet="pract7sharding"
mongod --shardsvr --port=1130 --dbpath=" C:\data\shard2" -replSet="pract7sharding"
mongod --shardsvr --port=1130 --dbpath=" C:\data\shard3" -replSet="pract7sharding"

 # Connect using mongosh mongosh --
host="localhost:1130"

```
PS C:\Users\DELL> mongosh --host="localhost:1130"
Current Mongosh Log ID: 67bc59a4b8aa48af10fa4213
Connecting to:          mongodb://localhost:1130/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.4.0
Using MongoDB:          7.0.16
Using Mongosh:          2.4.0

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

------
   The server generated these startup warnings when booting
   2025-02-24T17:02:26.339+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestric
   2025-02-24T17:02:26.340+05:30: This server is bound to localhost. Remote systems will be unable to connect to this server. Start the server
 --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable this warning
------
```

```
rs.initiate({
  _id:"pract7sharding",     members: [
    {_id: 0, host: "localhost:1130"},
    {_id: 1, host: "localhost:1140"},
    {_id: 2, host: "localhost:1150"}
  ]
})
```

```
test> rs.initiate({
...      _id:"pract7sharding",
...      members: [
...          {_id: 0, host: "localhost:1130"},
...          {_id: 1, host: "localhost:1140"},
...          {_id: 2, host: "localhost:1150"}
...      ]
... })
{ ok: 1 }
pract7sharding [direct: other] test>
```

Initialize a Query Router which is a mongos process.

mongos --port=1210 --
configdb="pract7/localhost:1030,localhost:1040,localhost:1050"

Now, Connect Shards and Query Router (mongos)

 # Connect to 'mongos' using 'mongosh'
mongosh --host="localhost:1210"

```
PS C:\Users\DELL> mongosh --host="localhost:1210"
Current Mongosh Log ID: 67bc5b6a198355841cfa4213
Connecting to:          mongodb://localhost:1210/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.4.0
Using MongoDB:          7.0.16
Using Mongosh:          2.4.0

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

------
   The server generated these startup warnings when booting
   2025-02-24T17:11:44.111+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
   2025-02-24T17:11:44.112+05:30: This server is bound to localhost. Remote systems will be unable to connect to this server. Start the server with
 --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with --bind_ip 127.0.0.1 to disable this warning
------
```

 # Add Shard Replica Set to Sharded Cluster
sh.addShard("pract7sharding/localhost:1130,localhost:1140,localhost:1150")

```
[direct: mongos] test> sh.addShard("pract7sharding/localhost:1130,localhost:1140,localhost:1150")
{
  shardAdded: 'pract7sharding',
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1740397806, i: 5 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1740397806, i: 5 })
}
```

Enable Sharding on a Specific Database of Shards Replica Set

sh.enableSharding("practice")

```
[direct: mongos] test> sh.enableSharding("practice")
{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1740397941, i: 8 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1740397941, i: 2 })
}
```

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: 1740398459, i: 17 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1740398459, i: 17 })
}
```

db.students.insertMany([{_id:1,name:"Riya",enroll:1997},{_id:2,name:"Prachi",enroll:2008},{
_id:3,name:"Neha",enroll:2002},{_id:4,name:"Manvi",enroll:"3000"}])

To Check how they are sorted

Sh.status()

```
[direct: mongos] practice> sh.status()
shardingVersion
{ _id: 1, clusterId: ObjectId('67bc51f2e35c803de22b667a') }
---
shards
[
  {
    _id: 'pract7sharding',
    host: 'pract7sharding/localhost:1130,localhost:1140,localhost:1150',
    state: 1,
    topologyTime: Timestamp({ t: 1740397806, i: 2 })
  }
]
---
active mongoses
[ { '7.0.16': 1 } ]
---
autosplit
{ 'Currently enabled': 'yes' }
---
balancer
{
  'Currently enabled': 'yes',
  'Currently running': 'no',
  'Failed balancer rounds in last 5 attempts': 0,
  'Migration Results for the last 24 hours': 'No recent migrations'
}
---
shardedDataDistribution
[
  {
    ns: 'config.system.sessions',
    shards: [
      {
        shardName: 'pract7sharding',
        numOrphanedDocs: 0,
        numOwnedDocuments: 9,
        ownedSizeBytes: 891,
        orphanedSizeBytes: 0
      }
    ]
  },
```

```
databases
[
  {
    database: { _id: 'config', primary: 'config', partitioned: true },
    collections: {
      'config.system.sessions': {
        shardKey: { _id: 1 },
        unique: false,
        balancing: true,
        chunkMetadata: [ { shard: 'pract7sharding', nChunks: 1 } ],
        chunks: [
          { min: { _id: MinKey() }, max: { _id: MaxKey() }, 'on shard': 'pract7sharding', 'last modified': Timestamp({ t: 1, i: 0 }) }
        ],
        tags: []
      }
    }
  },
  {
    database: {
      _id: 'practice',
      primary: 'pract7sharding',
      partitioned: false,
      version: {
        uuid: UUID('0425e83a-983c-45ad-986c-9c1e2bc86b01'),
        timestamp: Timestamp({ t: 1740397940, i: 1 }),
        lastMod: 1
      }
    },
    collections: {
      'practice.students': {
        shardKey: { enroll: 'hashed' },
        unique: false,
        balancing: true,
        chunkMetadata: [ { shard: 'pract7sharding', nChunks: 2 } ],
        chunks: [
          { min: { enroll: MinKey() }, max: { enroll: Long('0') }, 'on shard': 'pract7sharding', 'last modified': Timestamp({ t: 1, i: 0 }) },
          { min: { enroll: Long('0') }, max: { enroll: MaxKey() }, 'on shard': 'pract7sharding', 'last modified': Timestamp({ t: 1, i: 1 }) }
        ],
        tags: []
      }
    }
  }
]
```

To Find By enroll db.students.find({enroll:2008})

```
[direct: mongos] practice> db.students.find({enroll:2008})
[ { _id: 2, name: 'Prachi', enroll: 2008 } ]
```