

Practical -5

Aim - Implementation of Replicas in MongoDB.

Replication

Replication is the process of synchronizing data across multiple servers. Replication provides redundancy and increases data availability with multiple copies of data on different database servers. Replication protects a database from the loss of a single server. Replication also allows you to recover from hardware failure and service interruptions. With additional copies of the data, you can dedicate one to disaster recovery, reporting, or backup.

Why Replication?

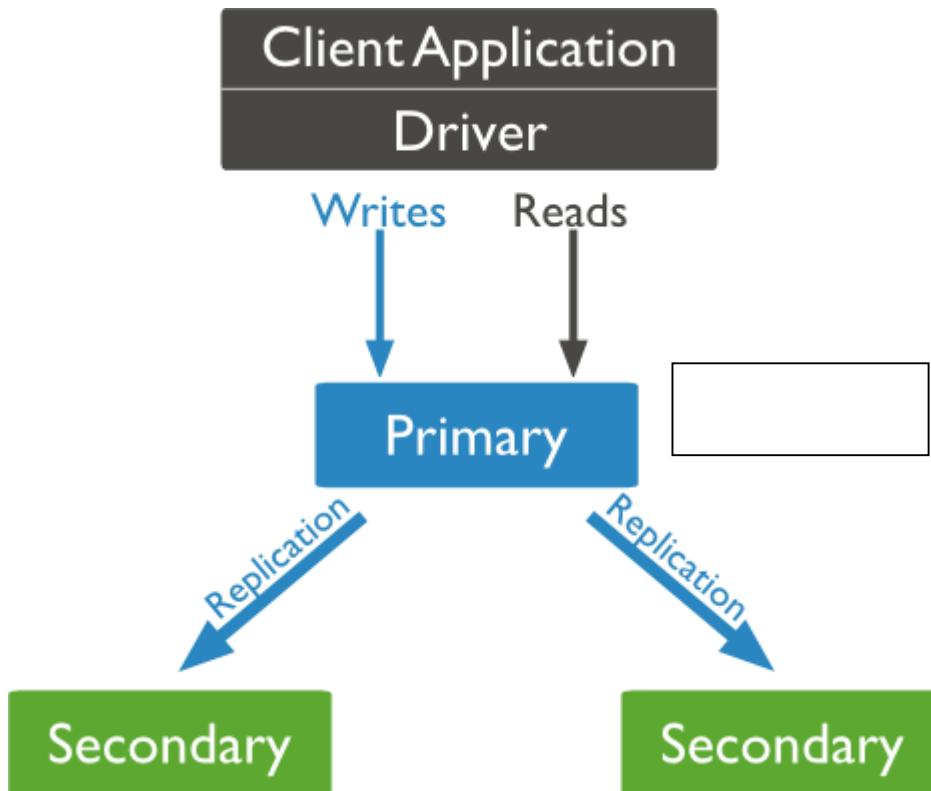
- To keep your data safe
- High (24*7) availability of data
- Disaster recovery
- No downtime for maintenance (like backups, index rebuilds, compaction)
- Read scaling (extra copies to read from)
- Replica set is transparent to the application

How Replication Works in MongoDB

MongoDB achieves replication by the use of replica set. A replica set is a group of **mongod** instances that host the same data set. In a replica, one node is primary node that receives all write operations. All other instances, such as secondaries, apply operations from the primary so that they have the same data set. Replica set can have only one primary node.

- Replica set is a group of two or more nodes (generally minimum 3 nodes are required).
- In a replica set, one node is primary node and remaining nodes are secondary.
- All data replicates from primary to secondary node.
- At the time of automatic failover or maintenance, election establishes for primary and a new primary node is elected.
- After the recovery of failed node, it again joins the replica set and works as a secondary node.

A typical diagram of MongoDB replication is shown in which client application always interact with the primary node and the primary node then replicates the data to the secondary nodes.



Replica Set Features

- A cluster of N nodes
- Any one node can be primary
- All write operations go to primary
- Automatic failover
- Automatic recovery
- Consensus election of primary

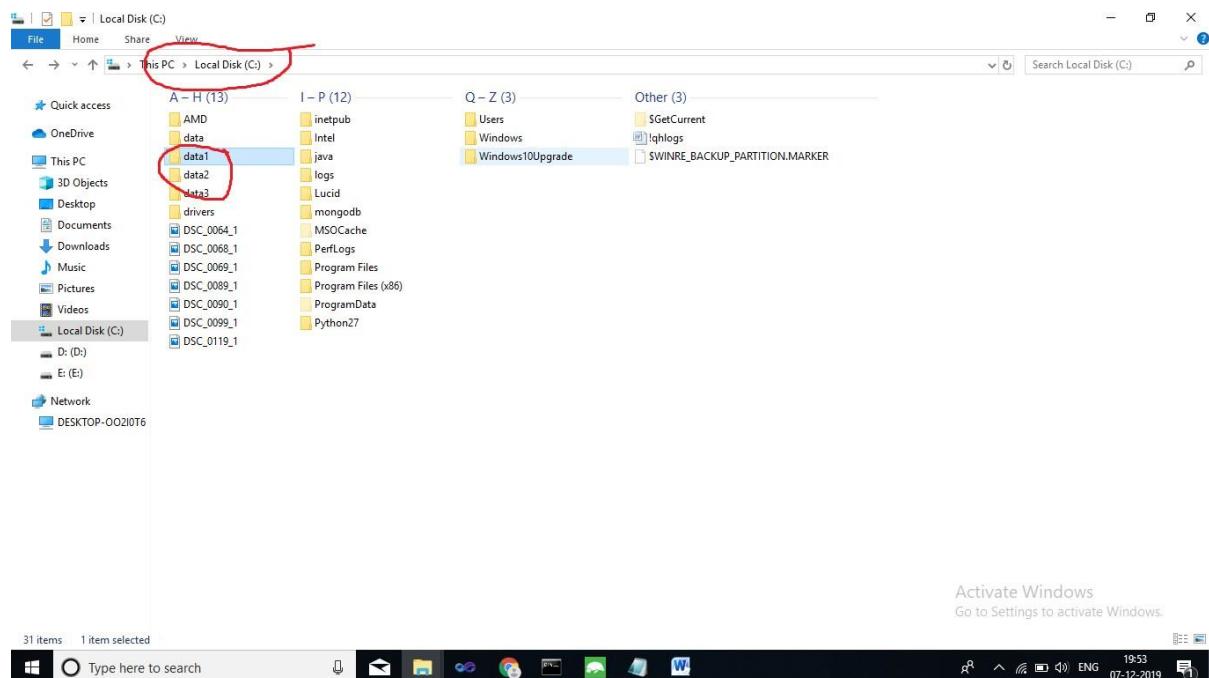
Example how to Set Up a Replica Set

Step 1

Shutdown already running MongoDB server.

Step 2

Create 3 folders named data1,data2,data3 at any location.Over here we have created in C drive.



- Over here we consider data1 is our primary server and data2 and data3 is replica.

Step 3(Creating Primary instance)

Start the MongoDB server by specifying -- replSet option. Following is the basic syntax.

```
mongod --port "PORT" --dbpath "YOUR_DB_DATA_PATH" --replSet
"REPLICA_SET_INSTANCE_NAME"
```

```
C:\Program Files\MongoDB\Server\4.2\bin>mongod --port 27018 --
dbpath "C:\data1" --replSet testrep
```

```
Command Prompt - mongod --port 27018 --dbpath "C:\data1" --replSet testrep
Microsoft Windows [Version 10.0.17134.523]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd C:\Program Files\MongoDB\Server\4.2\bin

:\Program Files\MongoDB\Server\4.2\bin>mongod --port 27018 --dbpath "C:\data1" --replSet testrep
2019-12-07T17:38:54.214+0530 I CONTROL [initandlisten] MongoDB starting: pid=16036 port=27018 dbpath=C:\data1 64-bit host=DESKTOP-002I0T6
2019-12-07T17:38:54.214+0530 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2019-12-07T17:38:54.215+0530 I CONTROL [initandlisten] git version: edfed45851c0b9ee15548f0f847df141764a317e
2019-12-07T17:38:54.215+0530 I CONTROL [initandlisten] allocator: tcmalloc
2019-12-07T17:38:54.215+0530 I CONTROL [initandlisten] modules: none
2019-12-07T17:38:54.216+0530 I CONTROL [initandlisten] build environment:
2019-12-07T17:38:54.216+0530 I CONTROL [initandlisten] distmod: 2012plus
2019-12-07T17:38:54.216+0530 I CONTROL [initandlisten] distarch: x86_64
2019-12-07T17:38:54.216+0530 I CONTROL [initandlisten] target arch: x86_64
2019-12-07T17:38:54.216+0530 I RECOVERY [initandlisten] WiredTiger recoveryTimestamp: Ts: Timestamp(0, 0)
2019-12-07T17:38:54.968+0530 I STORAGE [initandlisten] Timestamp monitor starting
2019-12-07T17:38:55.060+0530 I CONTROL [initandlisten]
2019-12-07T17:38:55.060+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2019-12-07T17:38:55.065+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2019-12-07T17:38:55.067+0530 I CONTROL [initandlisten]
2019-12-07T17:38:55.069+0530 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2019-12-07T17:38:55.070+0530 I CONTROL [initandlisten] ** Remote systems will be unable to connect to this server.
2019-12-07T17:38:55.072+0530 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP
2019-12-07T17:38:55.074+0530 I CONTROL [initandlisten] addresses it should serve responses from, or with --bind_ip all to
2019-12-07T17:38:55.076+0530 I CONTROL [initandlisten] bind to all interfaces. If this behavior is desired, start the
2019-12-07T17:38:55.078+0530 I CONTROL [initandlisten] server with --bind_ip 127.0.0.1 to disable this warning.
2019-12-07T17:38:55.080+0530 I CONTROL [initandlisten]
2019-12-07T17:38:55.102+0530 I SHARDING [initandlisten] Marking collection local.system.replset as collection version: <unsharded>
2019-12-07T17:38:55.103+0530 I STORAGE [initandlisten] Flow Control is enabled on this deployment.
2019-12-07T17:38:55.105+0530 I SHARDING [initandlisten] Marking collection admin.system.roles as collection version: <unsharded>
2019-12-07T17:38:55.109+0530 I SHARDING [initandlisten] Marking collection admin.system.version as collection version: <unsharded>
2019-12-07T17:38:55.111+0530 I STORAGE [initandlisten] createCollection: local.startup_log with generated UUID: 045bdff6-d9a5-45e1-93f5-b0e114f55e4c and options: { ca
pped: true, size: 10485760 }
2019-12-07T17:38:55.262+0530 I INDEX [initandlisten] index build: done building index _id_ on ns local.startup_log
2019-12-07T17:38:55.262+0530 I SHARDING [initandlisten] Marking collection local.startup_log as collection version: <unsharded>
```

Activate Windows
Go to Settings to activate Windows.

Now Open another command prompt for client. We will use this window to query our first server instance

C:\>mongo --port 27018

```
Command Prompt - mongo --port 27018
Microsoft Windows [Version 10.0.17134.523]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd

:\>mongo --port 27018
MongoDB shell version: v4.2.1
connecting to: mongodb://127.0.0.1:27018/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UID("41172f3b-8595-45e6-8ce9-794c18dce79a") }
MongoDB server version: 4.2.1
Server has startup warnings:
2019-12-07T17:38:55.060+0530 I CONTROL [initandlisten]
2019-12-07T17:38:55.060+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2019-12-07T17:38:55.065+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2019-12-07T17:38:55.067+0530 I CONTROL [initandlisten]
2019-12-07T17:38:55.069+0530 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2019-12-07T17:38:55.070+0530 I CONTROL [initandlisten] ** Remote systems will be unable to connect to this server.
2019-12-07T17:38:55.072+0530 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP
2019-12-07T17:38:55.074+0530 I CONTROL [initandlisten] addresses it should serve responses from, or with --bind_ip all to
2019-12-07T17:38:55.076+0530 I CONTROL [initandlisten] bind to all interfaces. If this behavior is desired, start the
2019-12-07T17:38:55.078+0530 I CONTROL [initandlisten] server with --bind_ip 127.0.0.1 to disable this warning.
2019-12-07T17:38:55.080+0530 I CONTROL [initandlisten]
...
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---

> show dbs
2019-12-07T17:40:35.263+0530 E QUERY [js] uncaught exception: Error: listDatabases failed:
  "operationTime" : Timestamp(0, 0),
  "ok" : 0,
  "errmsg" : "not master and slaveOk=false",
  "code" : 13435,
  "codeName" : "NotMasterNoSlaveOk",
  "$clusterTime" : {
    "clusterTime" : Timestamp(0, 0),
    "signature" : {
```

Activate Windows
Go to Settings to activate Windows.

Now Primary server is working.

Step 4(Creating 1(first) Replicas of Primary instance)

Syntax

C:\Program Files\MongoDB\Server\4.2\bin>mongod --port 27019 --dbpath "C:\data2" --replSet testrep

```

C:\Command Prompt - mongod --port 27019 --dbpath "C:\data2" --replSet testrep
Microsoft Windows [Version 10.0.17134.523]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd\

C:\>cd Program Files\MongoDB\Server\4.2\bin

C:\Program Files\MongoDB\Server\4.2\bin>mongod --port 27019 --dbpath "C:\data2" --replSet testrep
2019-12-07T18:28:26.326+0530 I CONTROL [initandlisten] MongoDB starting: pid=6328 port=27019 dbpath=C:\data2 64-bit host=DESKTOP-00210E
2019-12-07T18:28:26.326+0530 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2019-12-07T18:28:26.327+0530 I CONTROL [initandlisten] git version: edffed45851c0b9ee15548f0f847df141764a317e
2019-12-07T18:28:26.327+0530 I CONTROL [initandlisten] allocator: tcmalloc
2019-12-07T18:28:26.328+0530 I CONTROL [initandlisten] modules: none
2019-12-07T18:28:26.328+0530 I CONTROL [initandlisten] build environment:
2019-12-07T18:28:26.328+0530 I CONTROL [initandlisten]   distmod: 2012plus
2019-12-07T18:28:26.328+0530 I CONTROL [initandlisten]   distarch: x86_64
2019-12-07T18:28:26.328+0530 I CONTROL [initandlisten]   target_arch: x86_64
2019-12-07T18:28:26.328+0530 I CONTROL [initandlisten] options: { net: { port: 27019 }, replication: { replSet: "testrep" }, storage: { dbPath: "C:\data2" } }
2019-12-07T18:28:26.331+0530 I STORAGE [initandlisten] wiredtiger_open config: create,cache_size=3532M,cache_overflow=(file_max=0M),session_max=33000,eviction=(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,archive=true,path=journal,compressor=snappy),file_manager=(close_idle_time=100000,close_scan_interval=10,close_handle_minimum=250),statistics_log=(wait=0),verbose=[recovery_progress,checkpoint_progress],
2019-12-07T18:28:26.331+0530 I STORAGE [initandlisten] WiredTiger message [15757233506:566574][6328:140724426128464], txn-recover: Set global recovery timestamp: (0,0)
2019-12-07T18:28:26.701+0530 I RECOVERY [initandlisten] WiredTiger recoveryTimestamp. Ts: Timestamp(0, 0)
2019-12-07T18:28:26.865+0530 I STORAGE [initandlisten] Timestamp monitor starting
2019-12-07T18:28:26.927+0530 I CONTROL [initandlisten]
2019-12-07T18:28:26.927+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2019-12-07T18:28:26.929+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2019-12-07T18:28:26.931+0530 I CONTROL [initandlisten]
2019-12-07T18:28:26.932+0530 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2019-12-07T18:28:26.932+0530 I CONTROL [initandlisten] ** Remote clients will be unable to connect to this server.
2019-12-07T18:28:26.933+0530 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP
2019-12-07T18:28:26.934+0530 I CONTROL [initandlisten] ** addresses it should serve responses from, or with --bind_ip_all to
2019-12-07T18:28:26.934+0530 I CONTROL [initandlisten] ** bind to all interfaces. If this behavior is desired, start the
2019-12-07T18:28:26.935+0530 I CONTROL [initandlisten] ** server with --bind_ip 127.0.0.1 to disable this warning.
2019-12-07T18:28:26.936+0530 I CONTROL [initandlisten]
2019-12-07T18:28:26.944+0530 I SHARDING [initandlisten] Marking collection local.system.replset as collection version: <unsharded>
2019-12-07T18:28:26.945+0530 I STORAGE [initandlisten] Flow Control is enabled on this deployment.
2019-12-07T18:28:26.946+0530 I SHARDING [initandlisten] Marking collection admin.system.roles as collection version: <unsharded>
2019-12-07T18:28:26.947+0530 I SHARDING [initandlisten] Marking collection admin.system.version as collection version: <unsharded>
2019-12-07T18:28:26.948+0530 I STORAGE [initandlisten] createCollection: local.startup_log with generated UUID: 9e00fbca-d225-46c3-a224-ed13c6c582c9 and options: { capped: true, size: 1048576 }

Windows Activation
Activate Windows
Go to Settings to activate Windows.
20:06
07-12-2019

```

Now Open another command prompt for client. We will use this window to query our second server instance

C:\>mongo --port 27019

```

C:\> mongo --port 27019
MongoDB shell version v4.2.1
connecting to: mongodb://127.0.0.1:27019/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("62cdb2c2-d76d-4f80-bc3d-3af4a54166b0") }
MongoDB server version: 4.2.1
Server has startup warnings:
2019-12-07T18:28:26.927+0530 I CONTROL [initandlisten]
2019-12-07T18:28:26.927+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2019-12-07T18:28:26.929+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2019-12-07T18:28:26.931+0530 I CONTROL [initandlisten]
2019-12-07T18:28:26.932+0530 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2019-12-07T18:28:26.932+0530 I CONTROL [initandlisten] ** Remote systems will be unable to connect to this server.
2019-12-07T18:28:26.933+0530 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP
addresses it should serve responses from, or 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.
2019-12-07T18:28:26.934+0530 I CONTROL [initandlisten] **
2019-12-07T18:28:26.936+0530 I CONTROL [initandlisten] **

Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
...
>
testrep:SECONDARY>
```

Step 5

Now go to the command prompt of Primary server's Client instance.

C:\>mongo --port 27018

```

C:\> mongo --port 27018
MongoDB shell version v4.2.1
connecting to: mongodb://127.0.0.1:27018/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("41172f3b-8595-45e6-8ce9-794c18dce79a") }
MongoDB server version: 4.2.1
Server has startup warnings:
2019-12-07T17:38:55.060+0530 I CONTROL [initandlisten]
2019-12-07T17:38:55.060+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2019-12-07T17:38:55.065+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2019-12-07T17:38:55.067+0530 I CONTROL [initandlisten]
2019-12-07T17:38:55.069+0530 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2019-12-07T17:38:55.070+0530 I CONTROL [initandlisten] ** Remote systems will be unable to connect to this server.
2019-12-07T17:38:55.072+0530 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP
addresses it should serve responses from, or 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.
2019-12-07T17:38:55.081+0530 I CONTROL [initandlisten] **
2019-12-07T17:38:55.083+0530 I CONTROL [initandlisten] **

Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
...
> show dbs
2019-12-07T17:40:35.263+0530 E QUERY [js] uncaught exception: Error: listDatabases failed:
{
  "operationTime" : Timestamp(0, 0),
  "ok" : 0,
  "errmsg" : "not master and slaveOk=false",
  "code" : 13435,
  "codeName" : "NotMasterNoSlaveOk",
  "$clusterTime" : {
    "clusterTime" : Timestamp(0, 0),
    "signature" : {
      "signature" : 0
    }
}
testrep:PRIMARY>
```

Now type the following code

```
config = { _id : "testrep" , members : [  
  {  
    _id : 0,  
    host : "localhost:27018"  
  }  
]  
}
```

Above code set id=0 to the first replica instance which is on port 27018.(PRIMARY INSTANCE)

```
ps Command Prompt - mongo --port 27018
Mongo.prototype.getDBs@src/mongo/shell/mongo.js:87:12
shellHelper.show@src/mongo/shell/utils.js:906:13
shellHelper@src/mongo/shell/utils.js:790:15
@(shellHelp2):1:1
> config =[
...   _id : "testrep", members :[
...     _id:0,
...     host:"localhost:27018"
...   ]
...
]
2019-12-07T17:45:30.712+0530 E QUERY    [js] uncaught exception: SyntaxError: missing ] after element list :
@(shell):3:3
> config =[ _id : "testrep", members :[ _id:0 , host:"localhost:27018" } ]]
2019-12-07T17:45:38.326+0530 E QUERY    [js] uncaught exception: SyntaxError: missing ] after element list :
@(shell):1:41
> config ={ _id : "testrep" , members :[
...   _id : 0,
...   host : "localhost:27018",
...
]
...
config ={ _id : "testrep" , members :[
...
{
... _id : 0,
... host : "localhost:27018"
...
]
...
}
{
  "_id" : "testrep",
  "members" : [
    {
      "_id" : 0,
      "host" : "localhost:27018"
    }
  ]
}
> rs.initiate(config)
{
  "ok" : 1,
  "$clusterTime" : {
    "clusterTime" : Timestamp(1575721063, 1),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}

Activate Windows
Go to Settings to activate Windows.

21:08 07-12-2019
```

After this write command

rs.initiate(config)

This command initiates a replica set with the current host as its only member. This is confirmed by the output, which should resemble the following:

```

Select Command Prompt - mongo --port 27018
[{"ok": 1, "$clusterTime": {"$clusterTime": Timestamp(1575721063, 1), "signature": {"hash": BinData(0, "AAAAAAAAAAAAAAAAAAAAAAA="), "keyId": NumberLong(0)}}, "operationTime": Timestamp(1575721063, 1)}, testrep:SECONDARY> rs.status();
{
  "set": "testrep",
  "date": ISODate("2019-12-07T12:18:31.945Z"),
  "myState": 1,
  "term": NumberLong(1),
  "syncingTo": "",
  "syncSourceHost": "",
  "syncSourceId": -1,
  "heartbeatIntervalMillis": NumberLong(2000),
  "majorityVoteCount": 1,
  "writeMajorityCount": 1,
  "optimes": {
    "lastCommittedOptime": {
      "ts": Timestamp(1575721106, 1),
      "t": NumberLong(1)
    },
    "lastCommittedWallTime": ISODate("2019-12-07T12:18:26.079Z"),
    "readConcernMajorityOptime": {
      "ts": Timestamp(1575721106, 1),
      "t": NumberLong(1)
    },
    "readConcernMajorityWallTime": ISODate("2019-12-07T12:18:26.079Z"),
    "appliedOptime": {
      "ts": Timestamp(1575721106, 1),
      "t": NumberLong(1)
    },
    "durableOptime": {
      "ts": Timestamp(1575721106, 1),
      "t": NumberLong(1)
    }
  }
}

```

Activate Windows
Go to Settings to activate Windows.

After this write write command

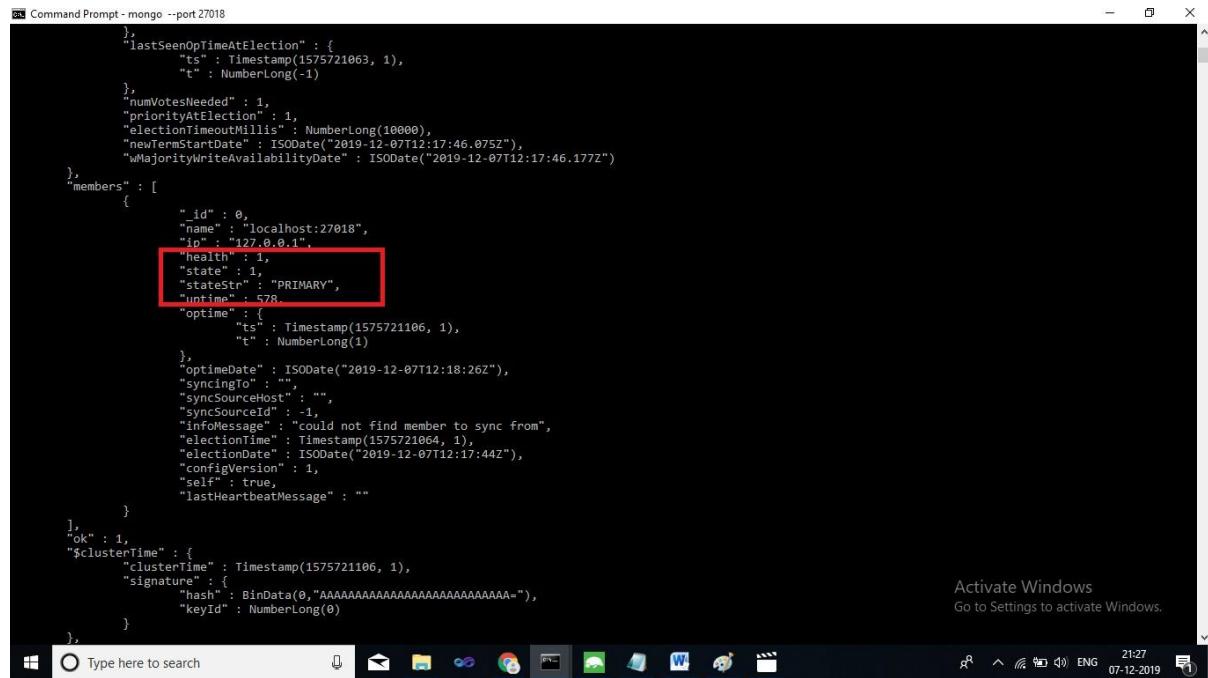
`rs.status()`

```

Command Prompt - mongo --port 27018
testrep:SECONDARY> rs.status();
{
  "set": "testrep",
  "date": ISODate("2019-12-07T12:18:31.945Z"),
  "myState": 1,
  "term": NumberLong(1),
  "syncingTo": "",
  "syncSourceHost": "",
  "syncSourceId": -1,
  "heartbeatIntervalMillis": NumberLong(2000),
  "majorityVoteCount": 1,
  "writeMajorityCount": 1,
  "optimes": {
    "lastCommittedOptime": {
      "ts": Timestamp(1575721106, 1),
      "t": NumberLong(1)
    },
    "lastCommittedWallTime": ISODate("2019-12-07T12:18:26.079Z"),
    "readConcernMajorityOptime": {
      "ts": Timestamp(1575721106, 1),
      "t": NumberLong(1)
    },
    "readConcernMajorityWallTime": ISODate("2019-12-07T12:18:26.079Z"),
    "appliedOptime": {
      "ts": Timestamp(1575721106, 1),
      "t": NumberLong(1)
    },
    "durableOptime": {
      "ts": Timestamp(1575721106, 1),
      "t": NumberLong(1)
    },
    "lastAppliedWallTime": ISODate("2019-12-07T12:18:26.079Z"),
    "lastDurableWallTime": ISODate("2019-12-07T12:18:26.079Z")
  },
  "lastStableRecoveryTimestamp": Timestamp(1575721066, 1),
  "lastStableCheckpointTimestamp": Timestamp(1575721066, 1),
  "electionCandidateMetrics": {
    "lastElectionReason": "electionTimeout",
    "lastElectionDate": ISODate("2019-12-07T12:17:44.209Z"),
    "termAtElection": NumberLong(1),
    "lastCommittedOptimeAtElection": {
      "ts": Timestamp(0, 0),
      "t": NumberLong(-1)
    }
  }
}

```

Activate Windows
Go to Settings to activate Windows.



```

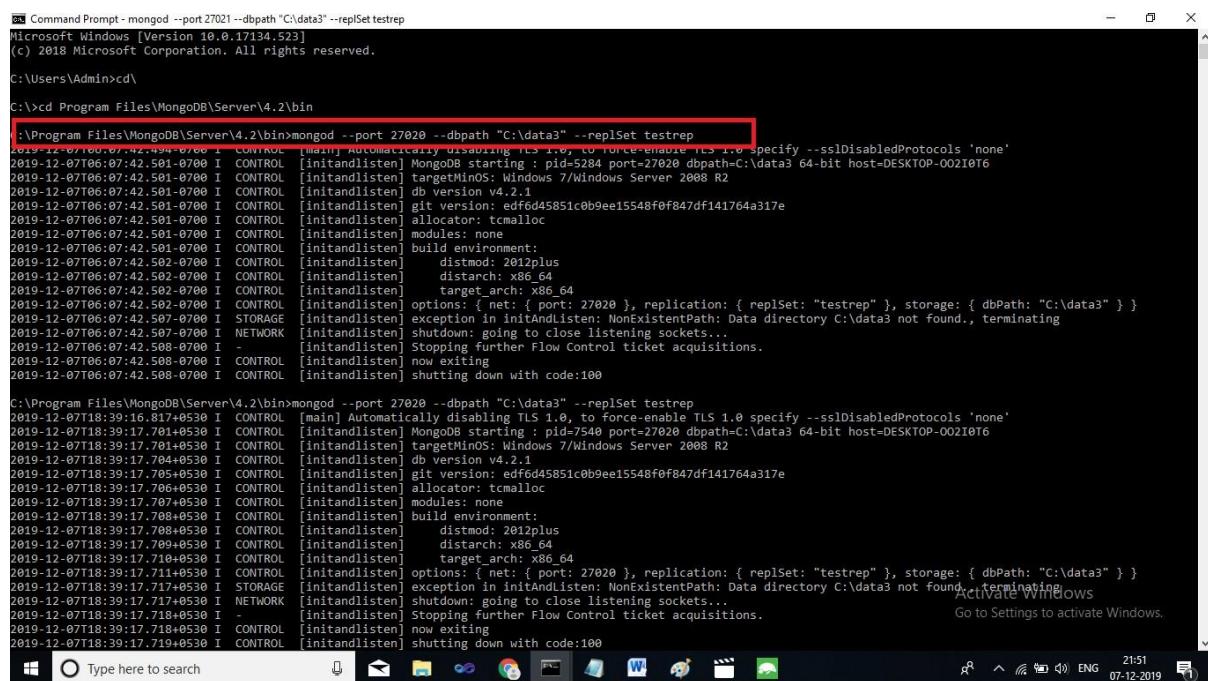
Command Prompt - mongo --port 27018
{
    "lastSeenOpTimeAtElection" : {
        "ts" : Timestamp(1575721063, 1),
        "t" : NumberLong(-1)
    },
    "numVotesNeeded" : 1,
    "priorityAtElection" : 1,
    "electionTimeoutMillis" : NumberLong(10000),
    "newTermStartDate" : ISODate("2019-12-07T12:17:46.075Z"),
    "wMajorityWriteAvailabilityDate" : ISODate("2019-12-07T12:17:46.177Z")
},
"members" : [
    {
        "_id" : 0,
        "name" : "localhost:27018",
        "ip" : "127.0.0.1",
        "health" : 1,
        "state" : 1,
        "stateStr" : "PRIMARY",
        "uptime" : 578,
        "optime" : {
            "ts" : Timestamp(1575721106, 1),
            "t" : NumberLong(1)
        },
        "optimeDate" : ISODate("2019-12-07T12:18:26Z"),
        "syncingTo" : "",
        "syncSourceHost" : "",
        "syncSourceId" : -1,
        "infoMessage" : "could not find member to sync from",
        "electionTime" : Timestamp(1575721064, 1),
        "electionDate" : ISODate("2019-12-07T12:17:44Z"),
        "configVersion" : 1,
        "self" : true,
        "lastHeartbeatMessage" : ""
    }
],
"ok" : 1,
"$clusterTime" : {
    "clusterTime" : Timestamp(1575721106, 1),
    "signature" : {
        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAA="),
        "keyId" : NumberLong(0)
    }
}
},
Activate Windows
Go to Settings to activate Windows.

```

Step 6(Creating 2(Second—port 27021) Replicas of Primary instance)

Syntax

C:\Program Files\MongoDB\Server\4.2\bin>mongod --port 27021 --dbpath "C:\data3" --repSet testrep



```

Command Prompt - mongod --port 27021 --dbpath "C:\data3" --repSet testrep
Microsoft Windows [Version 10.0.17134.523]
(c) 2018 Microsoft Corporation. All rights reserved.

C:>>>cd Program Files\MongoDB\Server\4.2\bin

:\>Program Files\MongoDB\Server\4.2\bin>mongod --port 27020 --dbpath "C:\data3" --repSet testrep
2019-12-07T06:07:42.434-0700 I CONTROL [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'
2019-12-07T06:07:42.501-0700 I CONTROL [initandlisten] MongoDB starting: pid=5284 port=27020 dbpath=C:\data3 64-bit host=DESKTOP-002I0T6
2019-12-07T06:07:42.501-0700 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2019-12-07T06:07:42.501-0700 I CONTROL [initandlisten] db version: v4.2.1
2019-12-07T06:07:42.501-0700 I CONTROL [initandlisten] git version: edfed45851c0b9ee15548f0f847df141764a317e
2019-12-07T06:07:42.501-0700 I CONTROL [initandlisten] allocator: tcmalloc
2019-12-07T06:07:42.501-0700 I CONTROL [initandlisten] modules: none
2019-12-07T06:07:42.501-0700 I CONTROL [initandlisten] build environment:
2019-12-07T06:07:42.502-0700 I CONTROL [initandlisten] distmod: 2012plus
2019-12-07T06:07:42.502-0700 I CONTROL [initandlisten] distarch: x86_64
2019-12-07T06:07:42.502-0700 I CONTROL [initandlisten] target arch: x86_64
2019-12-07T06:07:42.502-0700 I CONTROL [initandlisten] options: { net: { port: 27020 }, replication: { repSet: "testrep" }, storage: { dbPath: "C:\data3" } }
2019-12-07T06:07:42.507-0700 I STORAGE [initandlisten] exception in initAndListen: NonExistentPath: Data directory C:\data3 not found., terminating
2019-12-07T06:07:42.507-0700 I NETWORK [initandlisten] shutdown: going to close listening sockets...
2019-12-07T06:07:42.508-0700 I - [initandlisten] Stopping further Flow Control ticket acquisitions.
2019-12-07T06:07:42.508-0700 I CONTROL [initandlisten] now exiting
2019-12-07T06:07:42.508-0700 I CONTROL [initandlisten] shutting down with code:100

C:\>Program Files\MongoDB\Server\4.2\bin>mongod --port 27020 --dbpath "C:\data3" --repSet testrep
2019-12-07T10:39:16.817+0530 I CONTROL [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'
2019-12-07T10:39:17.781+0530 I CONTROL [initandlisten] MongoDB starting: pid=7540 port=27020 dbpath=C:\data3 64-bit host=DESKTOP-002I0T6
2019-12-07T10:39:17.781+0530 I CONTROL [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2019-12-07T10:39:17.784+0530 I CONTROL [initandlisten] db version: v4.2.1
2019-12-07T10:39:17.785+0530 I CONTROL [initandlisten] git version: edfed45851c0b9ee15548f0f847df141764a317e
2019-12-07T10:39:17.786+0530 I CONTROL [initandlisten] allocator: tcmalloc
2019-12-07T10:39:17.787+0530 I CONTROL [initandlisten] modules: none
2019-12-07T10:39:17.788+0530 I CONTROL [initandlisten] build environment:
2019-12-07T10:39:17.788+0530 I CONTROL [initandlisten] distmod: 2012plus
2019-12-07T10:39:17.789+0530 I CONTROL [initandlisten] distarch: x86_64
2019-12-07T10:39:17.710+0530 I CONTROL [initandlisten] target_arch: x86_64
2019-12-07T10:39:17.711+0530 I CONTROL [initandlisten] options: { net: { port: 27020 }, replication: { repSet: "testrep" }, storage: { dbPath: "C:\data3" } }
2019-12-07T10:39:17.717+0530 I STORAGE [initandlisten] exception in initAndListen: NonExistentPath: Data directory C:\data3 not found., terminating
2019-12-07T10:39:17.717+0530 I NETWORK [initandlisten] shutdown: going to close listening sockets...
2019-12-07T10:39:17.718+0530 I - [initandlisten] Stopping further Flow Control ticket acquisitions.
2019-12-07T10:39:17.718+0530 I CONTROL [initandlisten] now exiting
2019-12-07T10:39:17.719+0530 I CONTROL [initandlisten] shutting down with code:100

```

After this start mongo client of 2 seconday instance.

```

C:\ Command Prompt - mongo --port 27021
: \ mongo --port 27021
MongoDB shell version v4.2.1
connecting to: mongodb://127.0.0.1:27021/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("443d74e1-b336-42e8-8aa6-022a60d3fc46") }
MongoDB server version: 4.2.1
Server has startup warnings:
2019-12-07T18:41:23.067+0530 I CONTROL [initandlisten] 
2019-12-07T18:41:23.067+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2019-12-07T18:41:23.071+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2019-12-07T18:41:23.074+0530 I CONTROL [initandlisten] 
2019-12-07T18:41:23.075+0530 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.
2019-12-07T18:41:23.076+0530 I CONTROL [initandlisten] ** Remote systems will be unable to connect to this server.
2019-12-07T18:41:23.078+0530 I CONTROL [initandlisten] ** Start the server with --bind_ip <address> to specify which IP
2019-12-07T18:41:23.079+0530 I CONTROL [initandlisten] ** addresses it should serve responses from, or with --bind_ip_all to
2019-12-07T18:41:23.085+0530 I CONTROL [initandlisten] ** bind to all interfaces. If this behavior is desired, start the
2019-12-07T18:41:23.088+0530 I CONTROL [initandlisten] ** server with --bind_ip 127.0.0.1 to disable this warning.
2019-12-07T18:41:23.089+0530 I CONTROL [initandlisten] 
...
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---

> ^C
bye

C:\>mongo --port 27021
MongoDB shell version v4.2.1
connecting to: mongodb://127.0.0.1:27021/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("0f33758d-75e3-4114-942e-958670423b1e") }
MongoDB server version: 4.2.1
Server has startup warnings:
2019-12-07T18:41:23.067+0530 I CONTROL [initandlisten] 
2019-12-07T18:41:23.067+0530 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2019-12-07T18:41:23.071+0530 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
2019-12-07T18:41:23.074+0530 I CONTROL [initandlisten] ** WARNING: This server is bound to localhost.

```

Step 7

Now goto the CMD Client window of Primary Client(port 27018)

Execute rs.add() method.

Syntax

rs.add("localhost:27021");

```
ok Command Prompt - mongo --port27018
and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()
To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
...
testrep:PRIMARY> rs.add("localhost:27021");
{
    "ok" : 1,
    "$clusterTime" : {
        "clusterTime" : Timestamp(1575725225, 1),
        "signature" : {
            "hash" : BinData(0, "AAAAAAAAAAAAAAAAAAAAAAA="),
            "keyId" : NumberLong(0)
        }
    },
    "operationTime" : Timestamp(1575725225, 1)
}
testrep:PRIMARY> rs.status();
{
    "set" : "testrep",
    "date" : ISODate("2019-12-07T13:27:57.090Z"),
    "myState" : 1,
    "term" : NumberLong(1),
    "syncingTo" : null,
    "syncSourceHost" : null,
    "syncSourceId" : -1,
    "heartbeatIntervalMillis" : NumberLong(2000),
    "majorityVoteCount" : 2,
    "writeMajorityCount" : 2,
    "optimes" : [
        {
            "lastCommittedOpTime" : {
                "ts" : Timestamp(1575725269, 1),
                "t" : NumberLong(1)
            },
            "lastCommittedWallTime" : ISODate("2019-12-07T13:27:49.636Z"),
            "readConcernMajorityOpTime" : {
                "ts" : Timestamp(1575725269, 1),
                "t" : NumberLong(1)
            },
            "readConcernMajorityWallTime" : ISODate("2019-12-07T13:27:49.636Z"),
            "appliedOpTime" : {
                "ts" : Timestamp(1575725269, 1),
                "t" : NumberLong(1)
            }
        }
    ]
}

Activate Windows
Go to Settings to activate Windows.
21:57
07-12-2019
```

After this execute rs.status command.

This will show status of our Cluster. In our cluster three instance first Primary and remaining two are replicas of primary instance.

```
estrep:PRIMARY> rs.status();
{
  "set" : "testrep",
  "date" : ISODate("2019-12-07T13:27:57.090Z"),
  "myState" : 1,
  "term" : NumberLong(1),
  "syncingTo" : "",
  "syncSourceHost" : "",
  "syncSourceId" : -1,
  "heartbeatIntervalMillis" : NumberLong(2000),
  "majorityVoteCount" : 2,
  "writeMajorityCount" : 2,
  "optimism" : {
    "lastCommittedOpTime" : {
      "ts" : Timestamp(1575725269, 1),
      "t" : NumberLong(1)
    },
    "lastCommittedWallTime" : ISODate("2019-12-07T13:27:49.636Z"),
    "readConcernMajorityOpTime" : {
      "ts" : Timestamp(1575725269, 1),
      "t" : NumberLong(1)
    },
    "readConcernMajorityWallTime" : ISODate("2019-12-07T13:27:49.636Z"),
    "appliedOpTime" : {
      "ts" : Timestamp(1575725269, 1),
      "t" : NumberLong(1)
    },
    "durableOpTime" : {
      "ts" : Timestamp(1575725269, 1),
      "t" : NumberLong(1)
    },
    "lastAppliedWallTime" : ISODate("2019-12-07T13:27:49.636Z"),
    "lastDurableWallTime" : ISODate("2019-12-07T13:27:49.636Z")
  },
  "lastStableRecoveryTimestamp" : Timestamp(1575725225, 1),
  "lastStableCheckpointTimestamp" : Timestamp(1575725225, 1),
  "electionCandidateMetrics" : {
    "lastSelectionReason" : "electionTimeout",
    "lastElectendDate" : ISODate("2019-12-07T12:17:44.209Z"),
    "termAtElection" : NumberLong(1),
    "lastCommittedOpTimeAtElection" : {
      "ts" : Timestamp(0, 0),
      "t" : NumberLong(-1)
    }
  }
}
Activate Windows
Go to Settings to activate Windows.

```

```

Command Prompt - mongo --port 27018
{
    "_id": 1,
    "name": "localhost:27019",
    "ip": "127.0.0.1",
    "health": 1,
    "state": 2,
    "stateStr": "SECONDARY",
    "uptime": 1664,
    "optime": {
        "ts": Timestamp(1575725269, 1),
        "t": NumberLong(1)
    },
    "optimeDurable": {
        "ts": Timestamp(1575725269, 1),
        "t": NumberLong(1)
    },
    "optimeDate": ISODate("2019-12-07T13:27:49Z"),
    "optimeDurableDate": ISODate("2019-12-07T13:27:49Z"),
    "lastHeartbeat": ISODate("2019-12-07T13:27:55.357Z"),
    "lastHeartbeatRecv": ISODate("2019-12-07T13:27:55.394Z"),
    "pingMs": NumberLong(0),
    "lastHeartbeatMessage": "",
    "syncingTo": "localhost:27018",
    "syncSourceHost": "localhost:27018",
    "syncSourceId": 0,
    "infoMessage": ""
},
{
    "_id": 2,
    "name": "localhost:27018",
    "ip": "127.0.0.1",
    "health": 1,
    "state": 2,
    "stateStr": "SECONDARY",
    "uptime": 1664,
    "optime": {
        "ts": Timestamp(1575725269, 1),
        "t": NumberLong(1)
    },
    "optimeDurable": {
        "ts": Timestamp(1575725269, 1),
        "t": NumberLong(1)
    },
    "optimeDate": ISODate("2019-12-07T13:27:49Z"),
    "optimeDurableDate": ISODate("2019-12-07T13:27:49Z"),
    "lastHeartbeat": ISODate("2019-12-07T13:27:55.357Z"),
    "lastHeartbeatRecv": ISODate("2019-12-07T13:27:55.394Z"),
    "pingMs": NumberLong(0),
    "lastHeartbeatMessage": "",
    "syncingTo": "localhost:27019",
    "syncSourceHost": "localhost:27019",
    "syncSourceId": 1,
    "infoMessage": ""
}

```

Activate Windows
Go to Settings to activate Windows.

```

Command Prompt - mongo --port 27018
{
    "_id": 1,
    "name": "localhost:27019",
    "ip": "127.0.0.1",
    "health": 1,
    "state": 2,
    "stateStr": "SECONDARY",
    "uptime": 1664,
    "optime": {
        "ts": Timestamp(1575725269, 1),
        "t": NumberLong(1)
    },
    "optimeDurable": {
        "ts": Timestamp(1575725269, 1),
        "t": NumberLong(1)
    },
    "optimeDate": ISODate("2019-12-07T13:27:49Z"),
    "optimeDurableDate": ISODate("2019-12-07T13:27:49Z"),
    "lastHeartbeat": ISODate("2019-12-07T13:27:55.357Z"),
    "lastHeartbeatRecv": ISODate("2019-12-07T13:27:55.394Z"),
    "pingMs": NumberLong(0),
    "lastHeartbeatMessage": "",
    "syncingTo": "localhost:27018",
    "syncSourceHost": "localhost:27018",
    "syncSourceId": 0,
    "infoMessage": "",
    "configVersion": 3
},
{
    "_id": 2,
    "name": "localhost:27018",
    "ip": "127.0.0.1",
    "health": 1,
    "state": 2,
    "stateStr": "SECONDARY",
    "uptime": 1664,
    "optime": {
        "ts": Timestamp(1575725269, 1),
        "t": NumberLong(1)
    },
    "optimeDurable": {
        "ts": Timestamp(1575725269, 1),
        "t": NumberLong(1)
    },
    "optimeDate": ISODate("2019-12-07T13:27:49Z"),
    "optimeDurableDate": ISODate("2019-12-07T13:27:49Z"),
    "lastHeartbeat": ISODate("2019-12-07T13:27:55.357Z"),
    "lastHeartbeatRecv": ISODate("2019-12-07T13:27:55.394Z"),
    "pingMs": NumberLong(0),
    "lastHeartbeatMessage": "",
    "syncingTo": "localhost:27019",
    "syncSourceHost": "localhost:27019",
    "syncSourceId": 1,
    "infoMessage": ""
},
{
    "_id": 3,
    "name": "localhost:27021",
    "ip": "127.0.0.1",
    "health": 1,
    "state": 2,
    "stateStr": "SECONDARY",
    "uptime": 51,
    "optime": {
        "ts": Timestamp(1575725269, 1),
        "t": NumberLong(1)
    },
    "optimeDurable": {
        "ts": Timestamp(1575725269, 1),
        "t": NumberLong(1)
    },
    "optimeDate": ISODate("2019-12-07T13:27:49Z"),
    "optimeDurableDate": ISODate("2019-12-07T13:27:49Z"),
    "lastHeartbeat": ISODate("2019-12-07T13:27:55.357Z"),
    "lastHeartbeatRecv": ISODate("2019-12-07T13:27:55.394Z"),
    "pingMs": NumberLong(0),
    "lastHeartbeatMessage": "",
    "syncingTo": "localhost:27019",
    "syncSourceHost": "localhost:27019",
    "syncSourceId": 2,
    "infoMessage": ""
}

```

Activate Windows
Go to Settings to activate Windows.

Step 8

On cmd window of(client of 27018 main) Add the replica of main instance which is created on local host port 27019 in Cluster using rs.add() method.

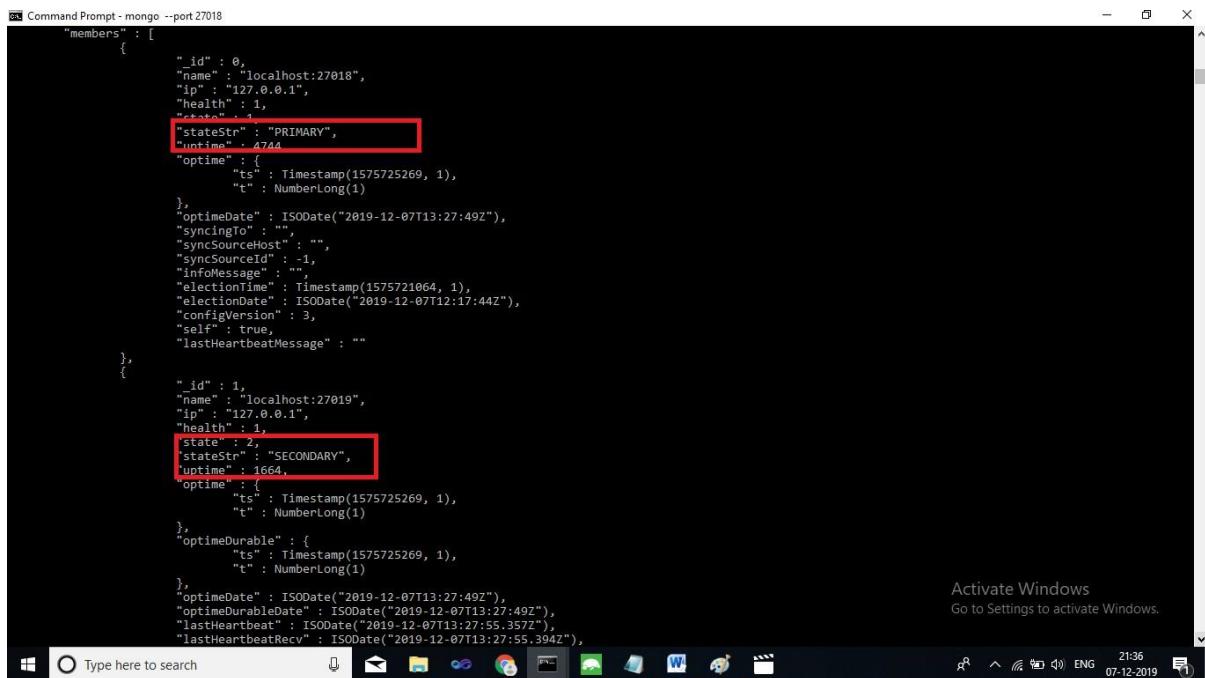
```
rs.add("localhost:27019");
```

```
ps Command Prompt - mongo --port 27018
testrep:PRIMARY> rs.add("localhost:27019");
{
    "_id" : 1,
    "$clusterTime" : {
        "clusterTime" : Timestamp(1575723613, 1),
        "signature" : {
            "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAA="),
            "keyId" : NumberLong(0)
        }
    },
    "operationTime" : Timestamp(1575723613, 1)
}
testrep:PRIMARY> rs.status();
{
    "set" : "testrep",
    "date" : ISODate("2019-12-07T13:00:47.638Z"),
    "myState" : 1,
    "term" : NumberLong(1),
    "syncingTo" : "",
    "syncSourceHost" : "",
    "syncSourceID" : -1,
    "heartbeatIntervalMillis" : NumberLong(2000),
    "majorityVoteCount" : 2,
    "writeMajorityCount" : 2,
    "optimes" : {
        "lastCommittedOptime" : {
            "ts" : Timestamp(1575723639, 1),
            "t" : NumberLong(1)
        },
        "lastCommittedWallTime" : ISODate("2019-12-07T13:00:39.454Z"),
        "readConcernMajorityOptime" : {
            "ts" : Timestamp(1575723639, 1),
            "t" : NumberLong(1)
        },
        "readConcernMajorityWallTime" : ISODate("2019-12-07T13:00:39.454Z"),
        "appliedOptime" : {
            "ts" : Timestamp(1575723639, 1),
            "t" : NumberLong(1)
        },
        "durableOptime" : {
            "ts" : Timestamp(1575723639, 1),
            "t" : NumberLong(1)
        },
    }
}
Activate Windows
Go to Settings to activate Windows.
21:33 07-12-2019
```

Now execute rs.status() command so we can see now in our cluster there are two instance one is primary and other is secondary.

```
estrep:PRIMARY> rs.status();
{
    "set" : "testrep",
    "date" : ISODate("2019-12-07T13:27:57.090Z"),
    "myState" : 1,
    "term" : NumberLong(1),
    "syncingTo" : "",
    "syncSourceHost" : "",
    "syncSourceId" : -1,
    "heartbeatIntervalMillis" : NumberLong(2000),
    "majorityVoteCount" : 2,
    "writeMajorityCount" : 2,
    "optimes" : {
        "lastCommittedOpTime" : {
            "ts" : Timestamp(1575725269, 1),
            "t" : NumberLong(1)
        },
        "lastCommittedWallTime" : ISODate("2019-12-07T13:27:49.636Z"),
        "readConcernMajorityOptime" : {
            "ts" : Timestamp(1575725269, 1),
            "t" : NumberLong(1)
        },
        "readConcernMajorityWallTime" : ISODate("2019-12-07T13:27:49.636Z"),
        "appliedOpTime" : {
            "ts" : Timestamp(1575725269, 1),
            "t" : NumberLong(1)
        },
        "durableOptime" : {
            "ts" : Timestamp(1575725269, 1),
            "t" : NumberLong(1)
        },
        "lastAppliedWallTime" : ISODate("2019-12-07T13:27:49.636Z"),
        "lastDurableWallTime" : ISODate("2019-12-07T13:27:49.636Z")
    },
    "lastStableRecoveryTimestamp" : Timestamp(1575725225, 1),
    "lastStableCheckpointTimestamp" : Timestamp(1575725225, 1),
    "electionCandidateMetrics" : {
        "lastElectionReason" : "electionTimeout",
        "lastElectionDate" : ISODate("2019-12-07T12:17:44.209Z"),
        "termAtElection" : NumberLong(1),
        "lastCommittedOpTimeAtElection" : {
            "ts" : Timestamp(0, 0),
            "t" : NumberLong(-1)
        }
    }
}

```



```

Command Prompt - mongo --port 27018
{
  "members" : [
    {
      "_id" : 0,
      "name" : "localhost:27018",
      "ip" : "127.0.0.1",
      "health" : 1,
      "state" : 1,
      "stateStr" : "PRIMARY",
      "uptime" : 4744,
      "optime" : {
        "ts" : Timestamp(1575725269, 1),
        "t" : NumberLong(1)
      },
      "optimeDate" : ISODate("2019-12-07T13:27:49Z"),
      "syncingTo" : "",
      "syncSourceHost" : "",
      "syncSourceId" : -1,
      "infoMessage" : "",
      "electionTime" : Timestamp(1575721064, 1),
      "electionDate" : ISODate("2019-12-07T12:17:44Z"),
      "configVersion" : 3,
      "self" : true,
      "lastHeartbeatMessage" : ""
    },
    {
      "_id" : 1,
      "name" : "localhost:27019",
      "ip" : "127.0.0.1",
      "health" : 1,
      "state" : 2,
      "stateStr" : "SECONDARY",
      "uptime" : 1664,
      "optime" : {
        "ts" : Timestamp(1575725269, 1),
        "t" : NumberLong(1)
      },
      "optimeDurable" : {
        "ts" : Timestamp(1575725269, 1),
        "t" : NumberLong(1)
      },
      "optimeDate" : ISODate("2019-12-07T13:27:49Z"),
      "optimeDurableDate" : ISODate("2019-12-07T13:27:49Z"),
      "lastHeartbeat" : ISODate("2019-12-07T13:27:55.357Z"),
      "lastHeartbeatRecv" : ISODate("2019-12-07T13:27:55.394Z")
    }
  ]
}

```

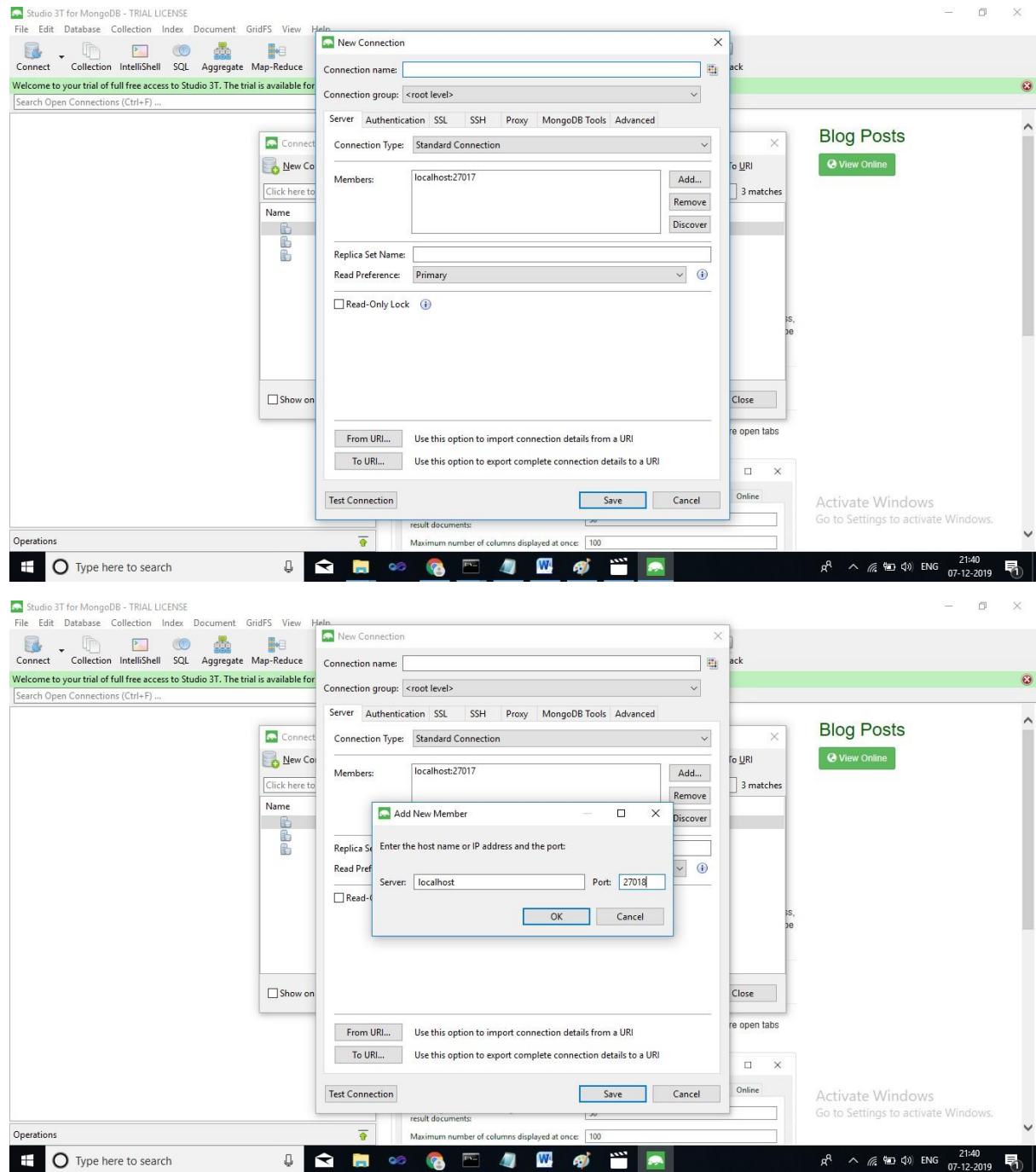
Activate Windows
Go to Settings to activate Windows.

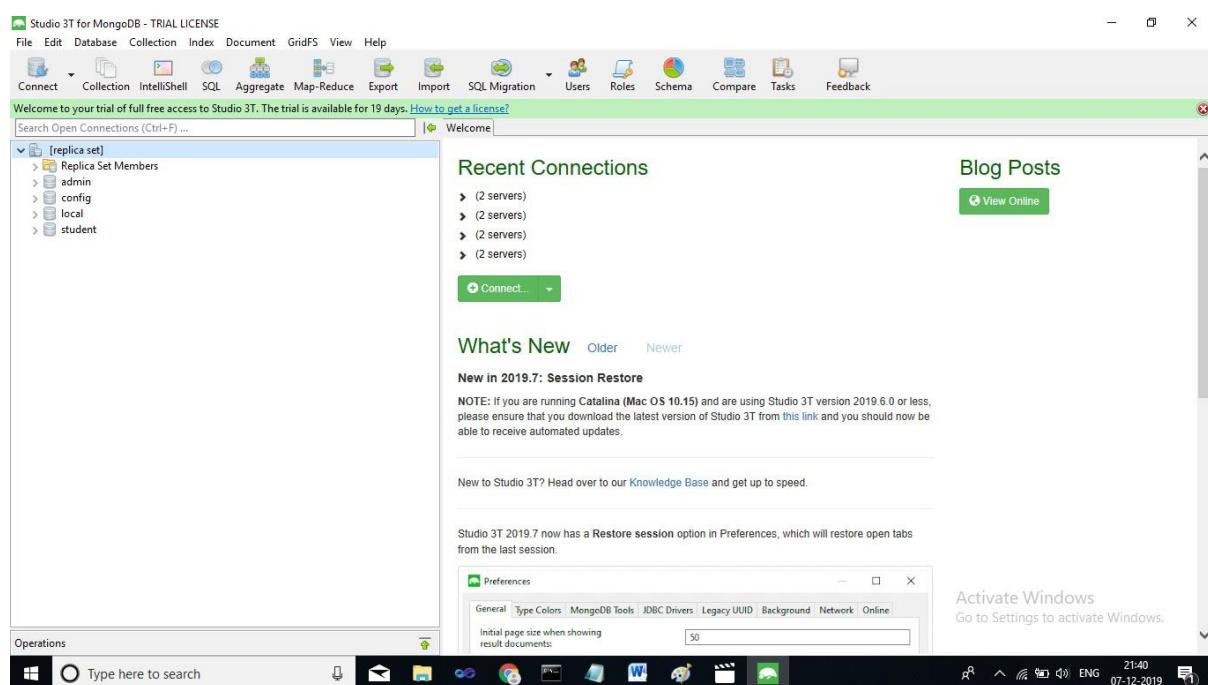
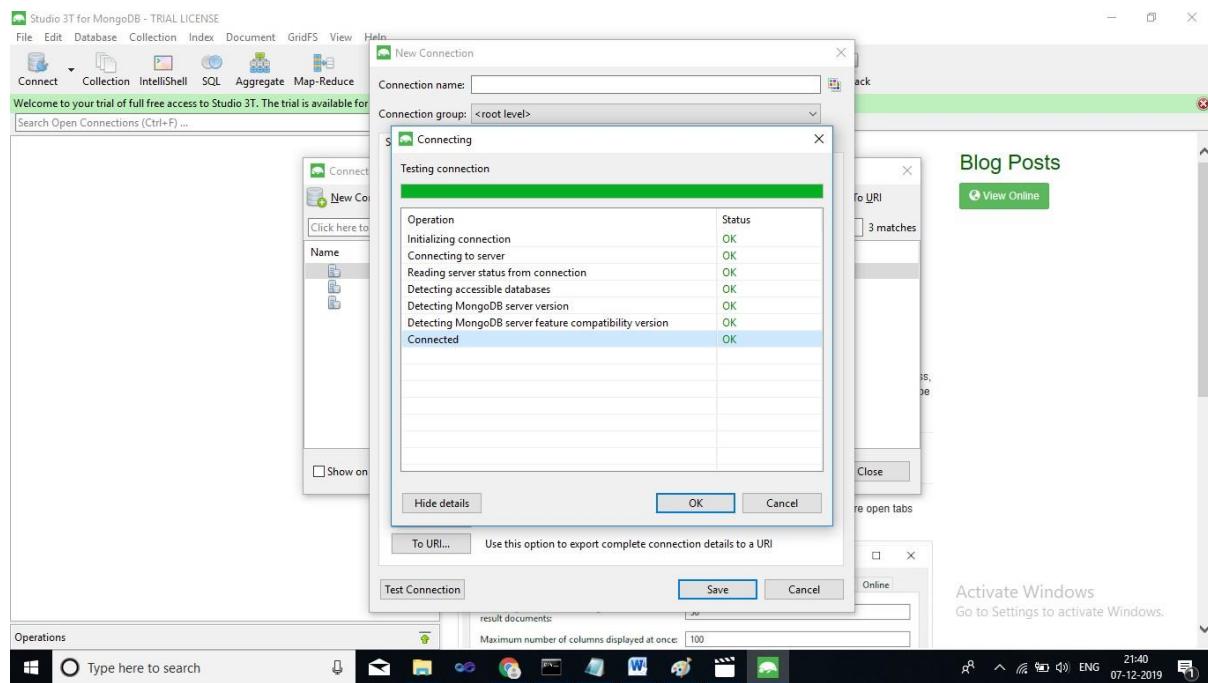
Step 9

Now OPEN studio 3T

Click on connect and add both 27018 Primary and 27019,27021 seconday servers. As given below.

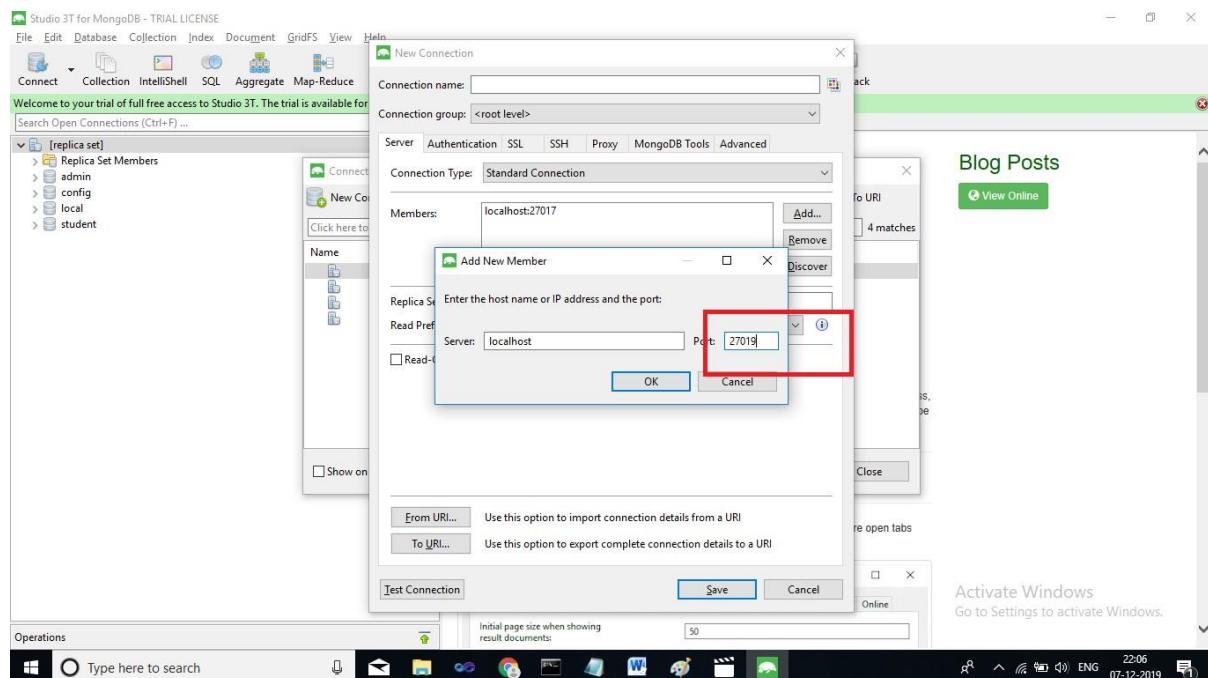
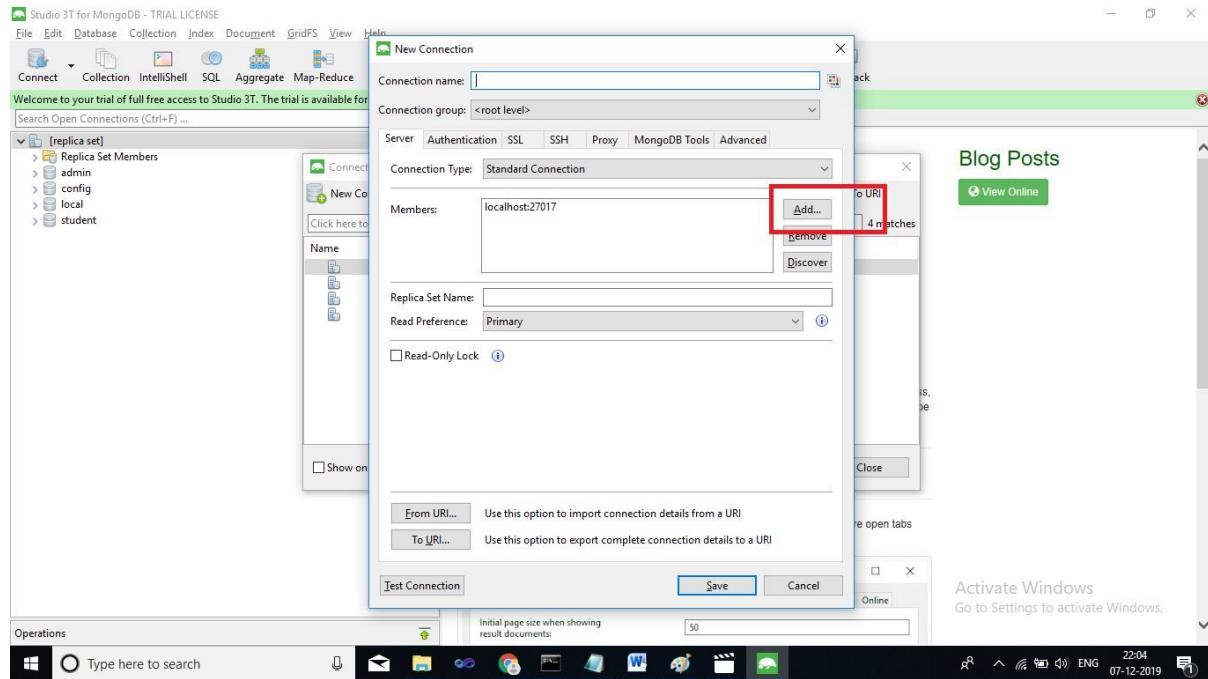
Connecting with primary instance 27018

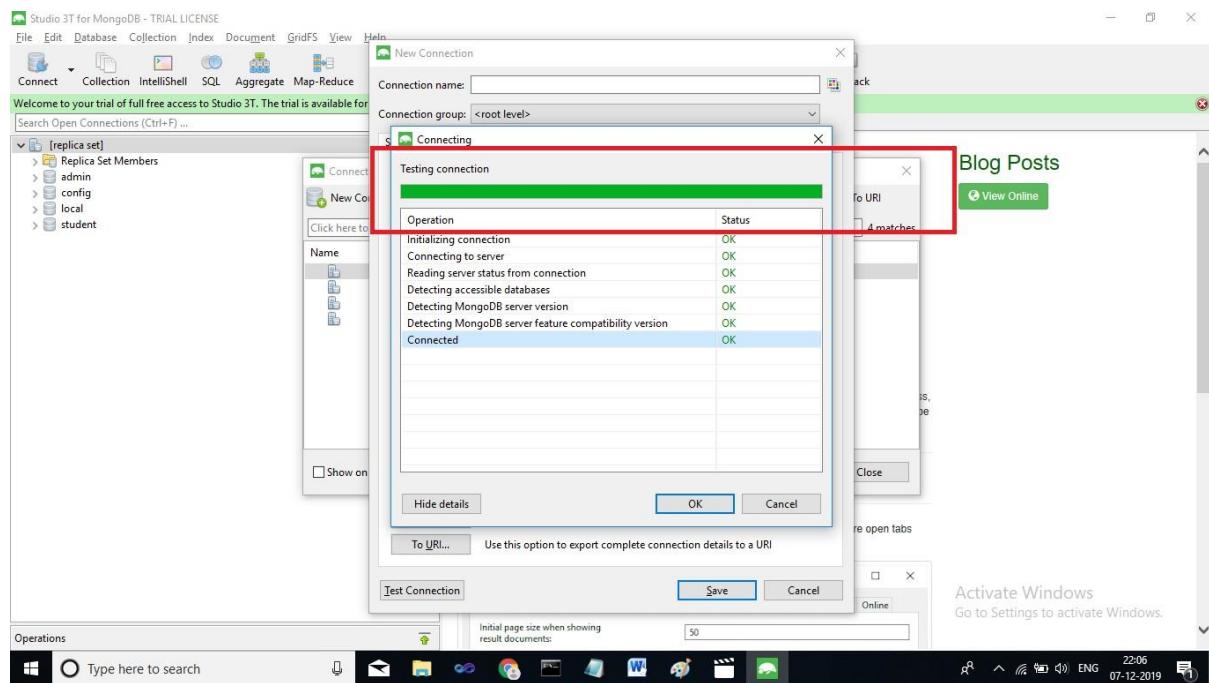




STEP 10

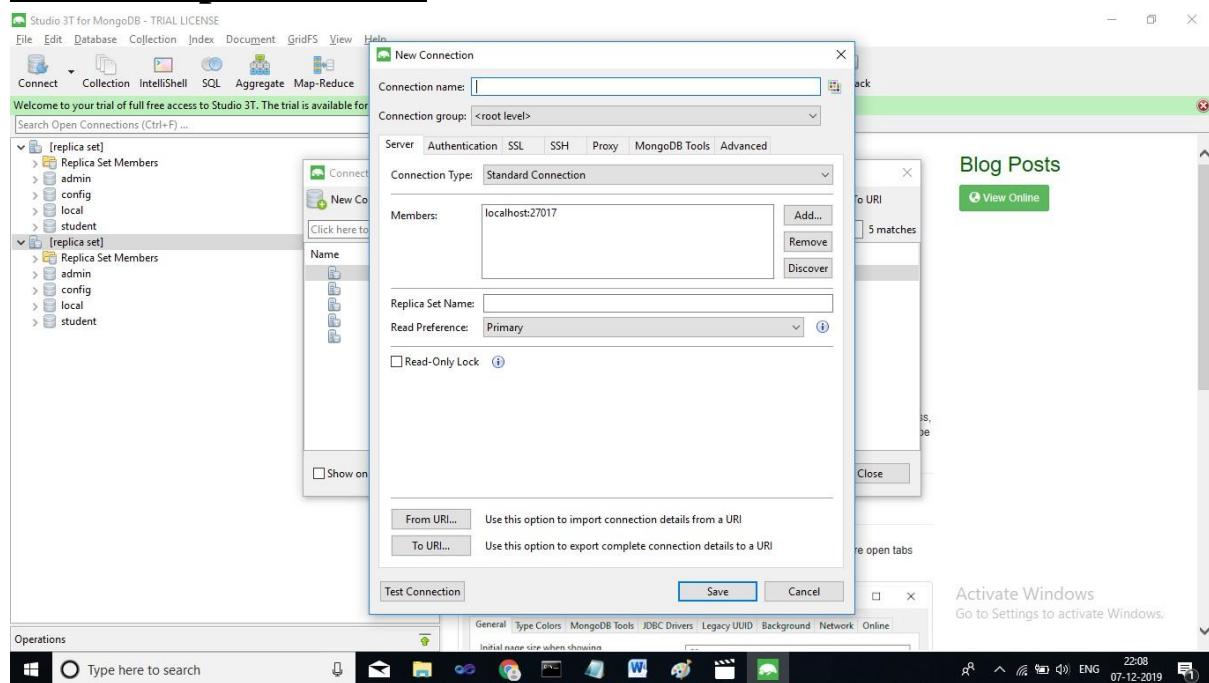
Add the 1(first) Secondary instance in Studio 3t. localhost port—27019

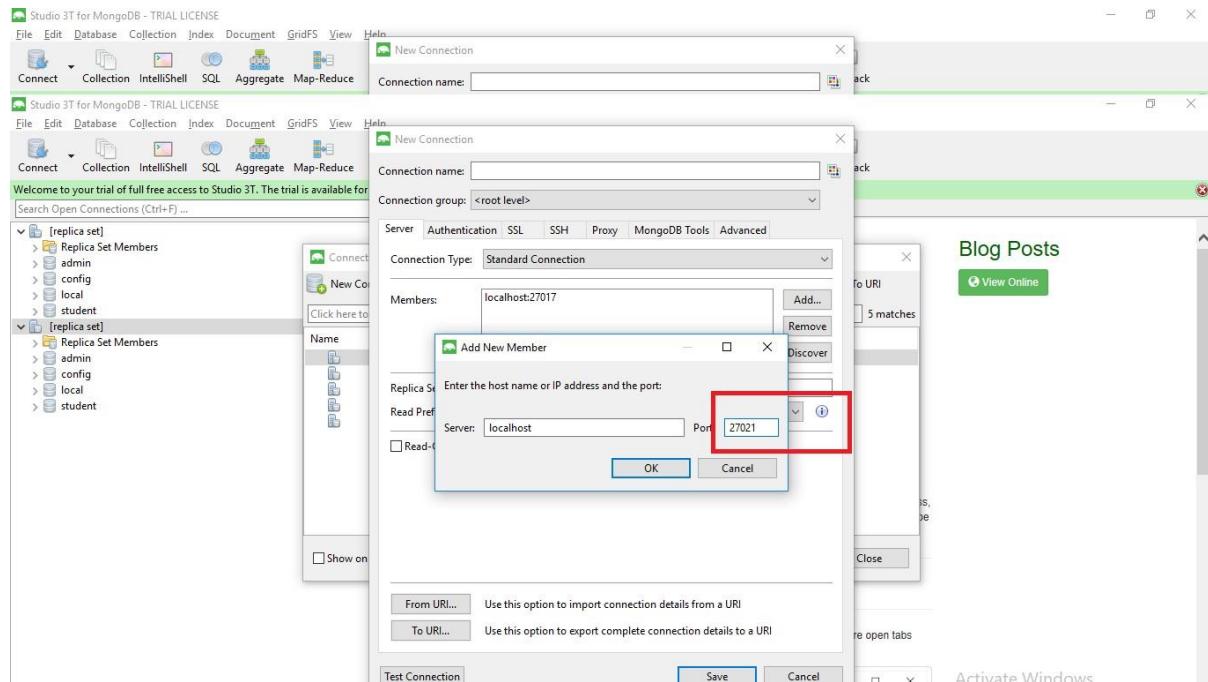




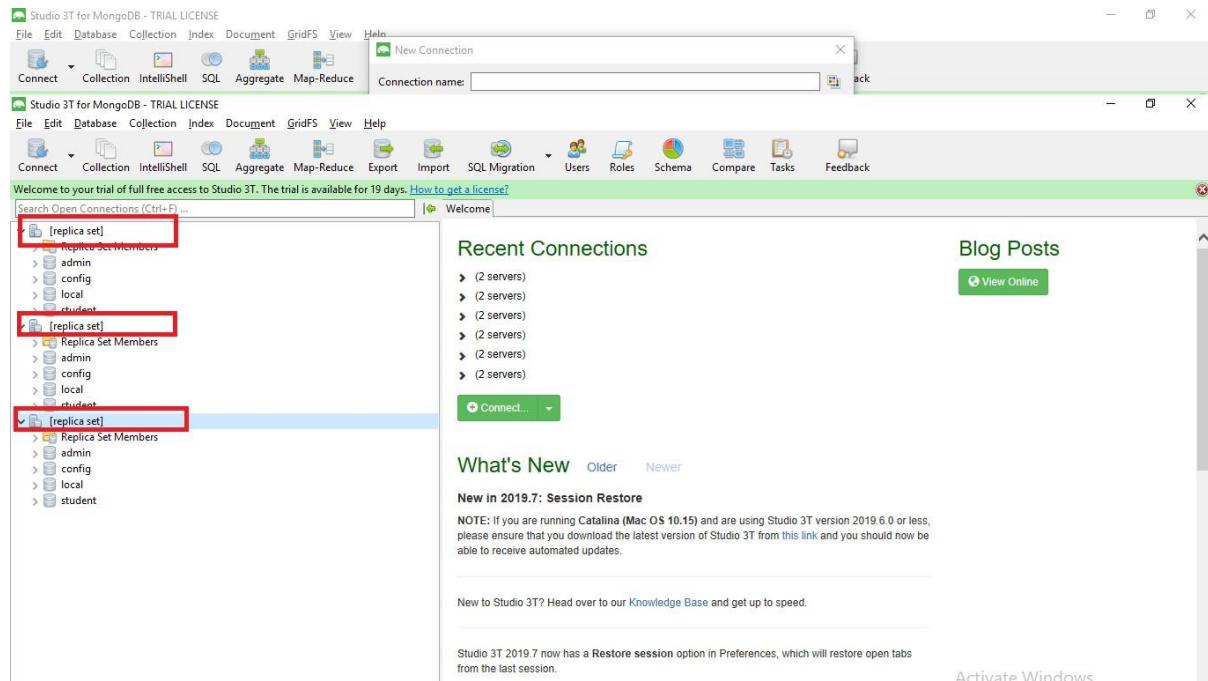
STEP 11

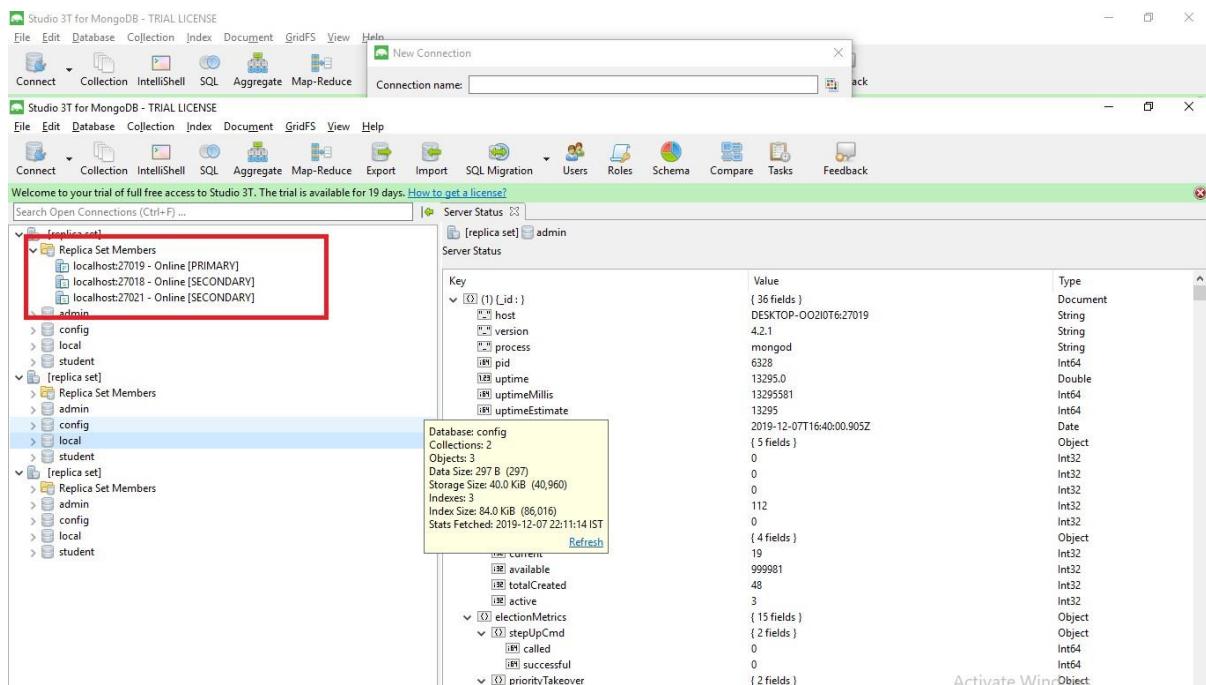
**Add the 2(second) Secondary instance in Studio 3t.
localhost port—27021**



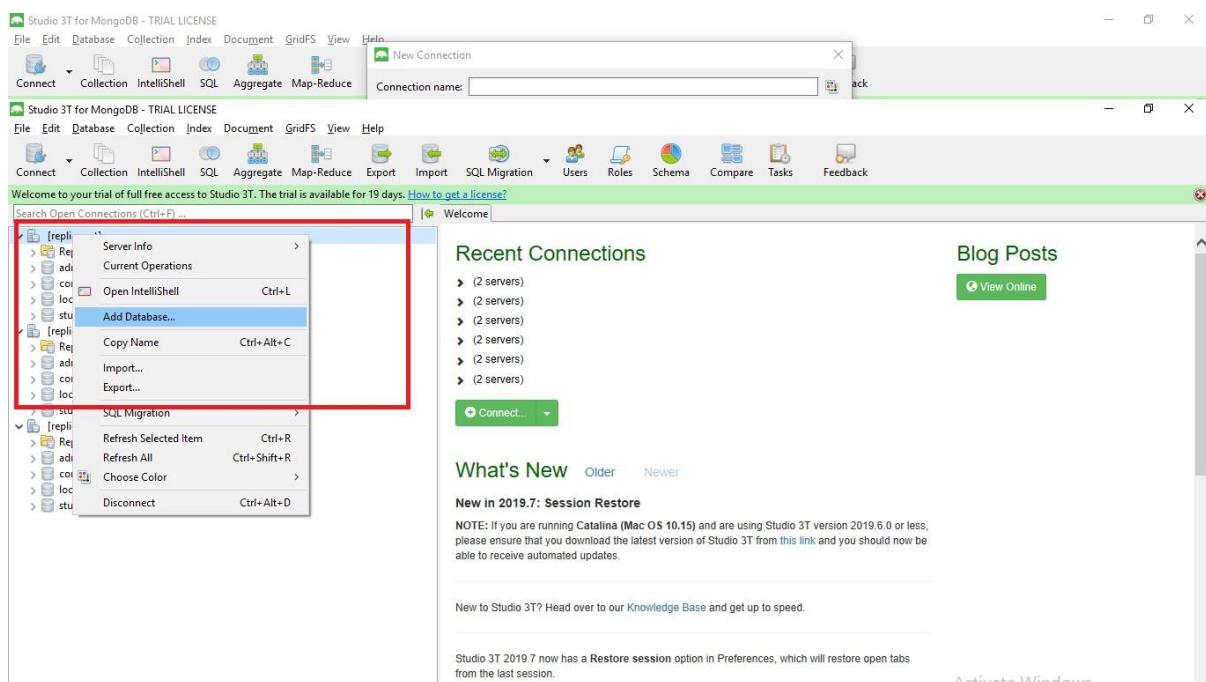


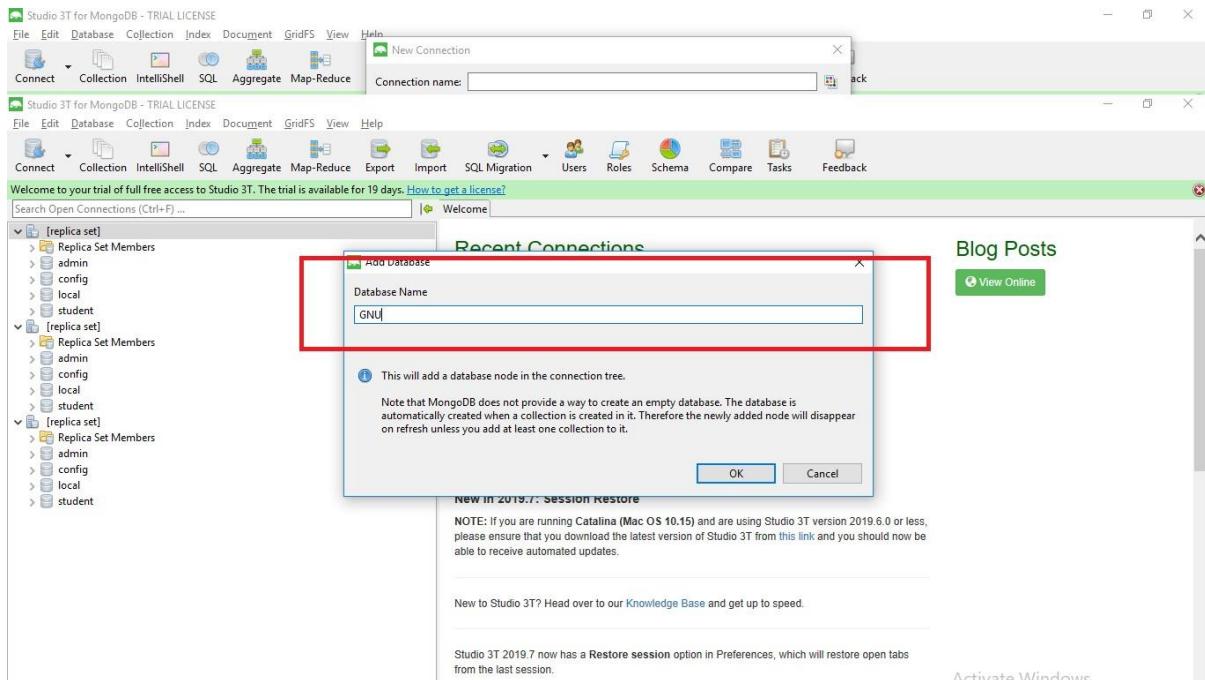
Now we can see we have 3 instance. One is Primary and other two are secondary.



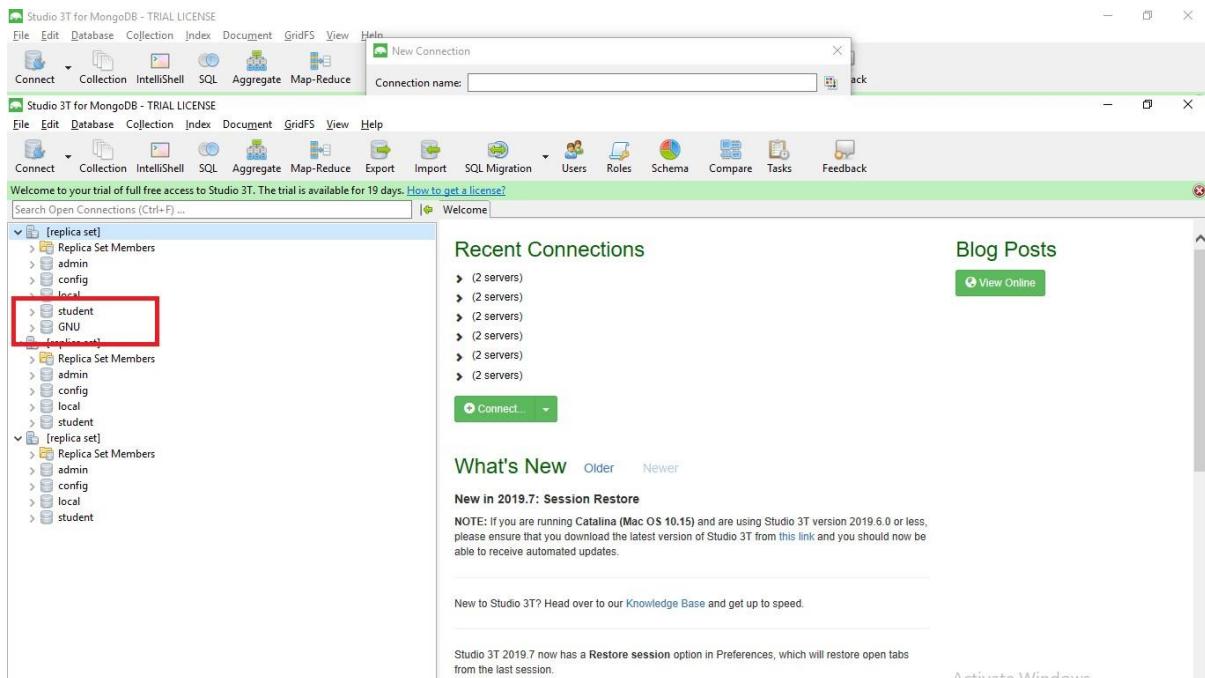


STEP 12. Create Database on Primary server.

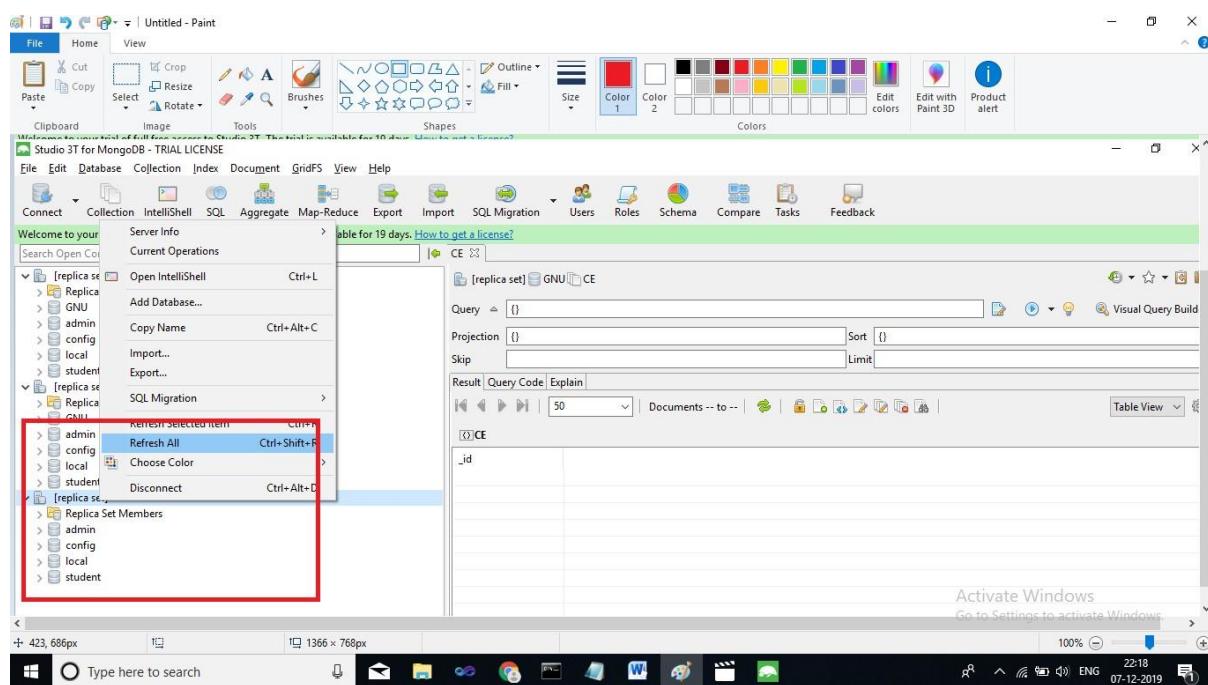
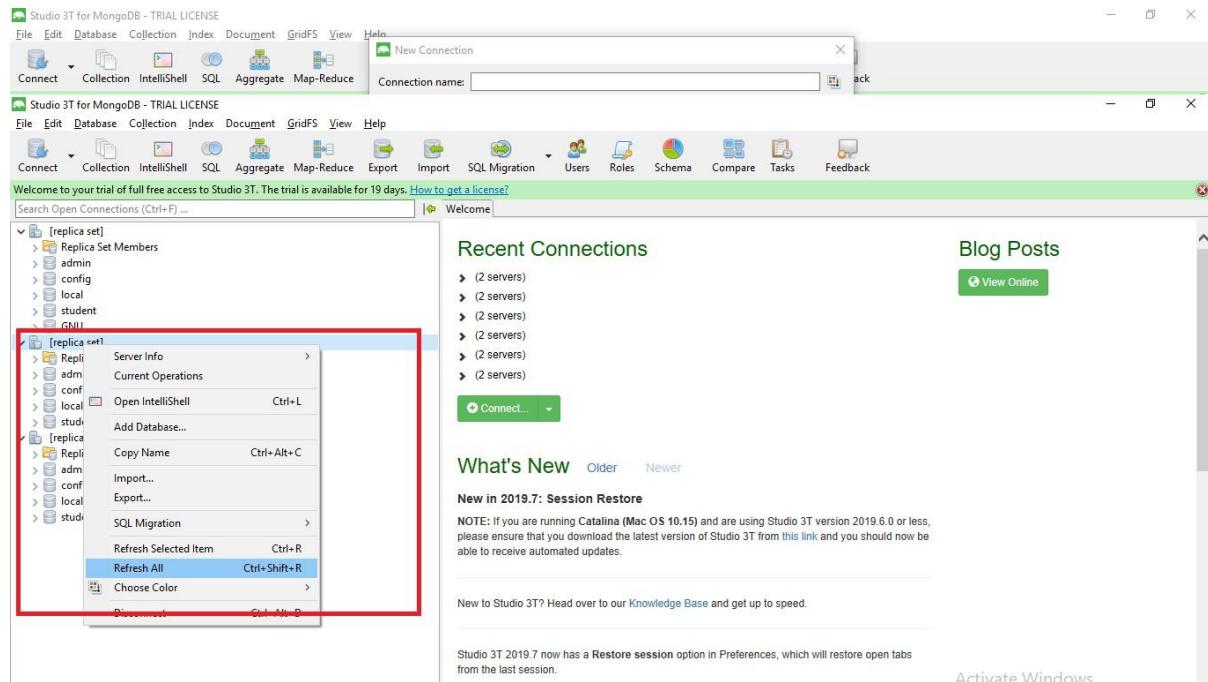




Database is created.—GNU



STEP 13 To Update Secondary Instances JUST PRESS REFRESH ALL OPTION.



STEP 14 To REMOVE ANY Secondary Instances.

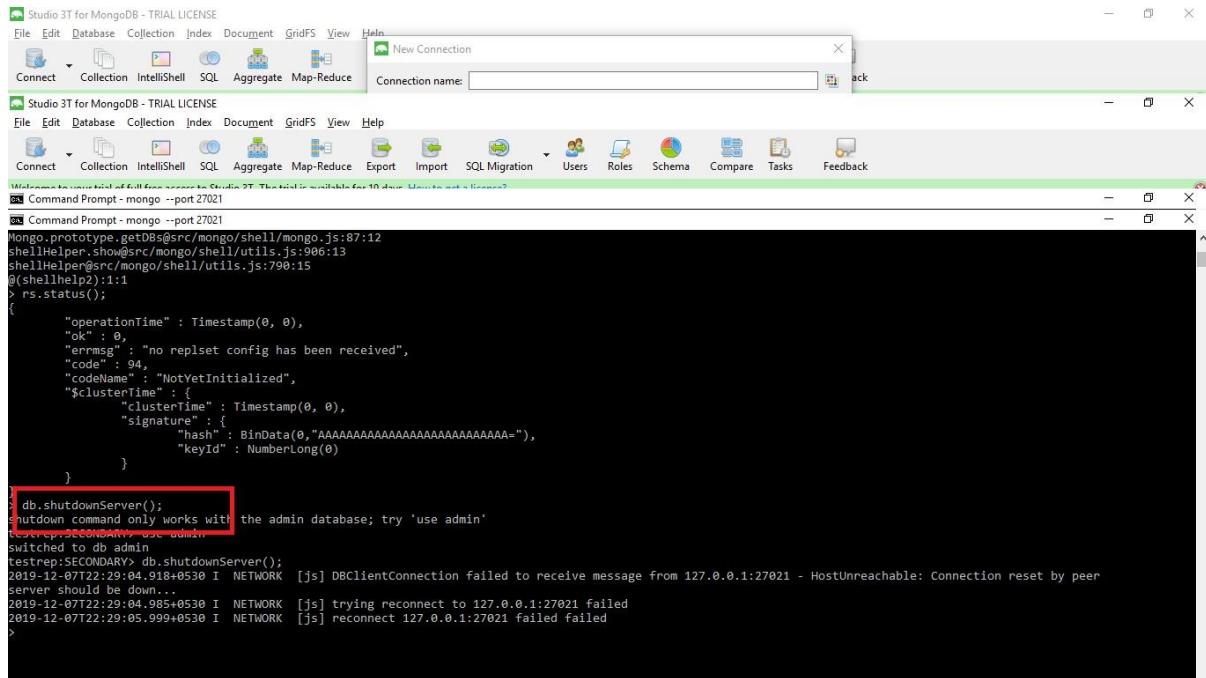
Shutdown the server which you want to remove.

Over here we remove second instance (port 27021)

From mongo client of port 27021

Syntax

```
db.shutdownServer();
```



```

Mongo.prototype.getDBs@src/mongo/shell/mongo.js:87:12
shellHelper.show@src/mongo/shell/utils.js:986:13
shellHelper@src/mongo/shell/utils.js:790:15
@shellHelp2):1:1
> rs.status();
{
  "operationTime" : Timestamp(0, 0),
  "ok" : 0,
  "errmsg" : "no replset config has been received",
  "code" : 94,
  "codeName" : "NotYetInitialized",
  "$clusterTime" : {
    "clusterTime" : Timestamp(0, 0),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
db.shutdownServer();
shutdown command only works with the admin database; try 'use admin'
testrep:SECONDARY> use admin
switched to db admin
testrep:SECONDARY> db.shutdownServer();
2019-12-07T22:29:04.918+0530 I NETWORK [js] DBClientConnection failed to receive message from 127.0.0.1:27021 - HostUnreachable: Connection reset by peer
server should be down...
2019-12-07T22:29:04.985+0530 I NETWORK [js] trying reconnect to 127.0.0.1:27021 failed
2019-12-07T22:29:05.999+0530 I NETWORK [js] reconnect 127.0.0.1:27021 failed
>

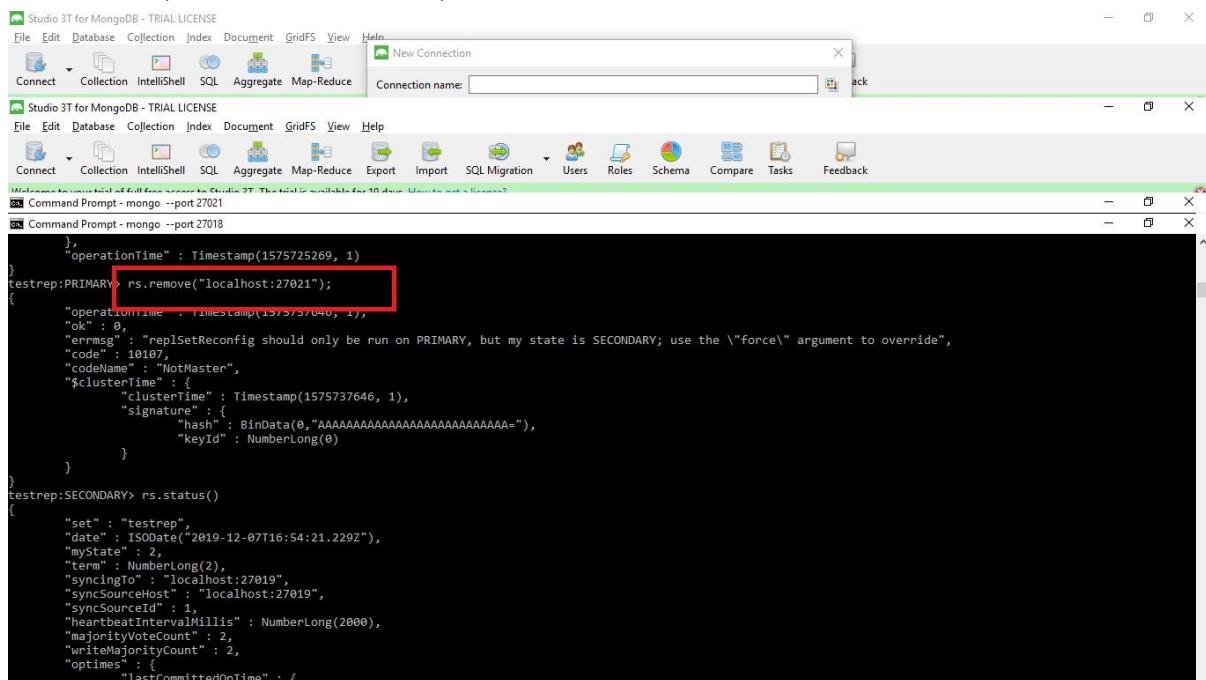
```

Now go to the CMD of primary client.(Port 27018)

And write rs.remove() method to remove secondary instance from cluster.

Syntax

`rs.remove("localhost:27021");`



```

testrep:PRIMARY> rs.remove("localhost:27021");
{
  "operationTime" : Timestamp(1575725269, 1)
}
testrep:SECONDARY> rs.status()
{
  "set" : "testrep",
  "date" : ISODate("2019-12-07T16:54:21.229Z"),
  "myState" : 2,
  "term" : NumberLong(2),
  "syncingTo" : "localhost:27019",
  "syncSourceHost" : "localhost:27019",
  "syncSourceId" : 1,
  "heartbeatIntervalMillis" : NumberLong(2000),
  "majorityVoteCount" : 2,
  "writeMajorityCount" : 2,
  "optimes" : {
    "lastCommittedOptime" : {
      "ts" : Timestamp(1575737640, 1),
      "ok" : 0,
      "errmsg" : "repSetReconfig should only be run on PRIMARY, but my state is SECONDARY; use the \"force\" argument to override",
      "code" : 10107,
      "codeName" : "NotMaster",
      "$clusterTime" : {
        "clusterTime" : Timestamp(1575737646, 1),
        "signature" : {
          "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAA="),
          "keyId" : NumberLong(0)
        }
      }
    }
}

```

Now check status of Cluster two know member of group.

Use command `rs.status();`

```

rsreplica:SECONDARY> rs.status()
{
    "version": 1,
    "date": ISODate("2019-12-07T16:54:21.229Z"),
    "myState": 2,
    "term": NumberLong(2),
    "syncingTo": "localhost:27019",
    "syncSourceHost": "localhost:27019",
    "syncSourceId": 1,
    "heartbeatIntervalMillis": NumberLong(2000),
    "majorityVoteCount": 2,
    "writeMajorityCount": 2,
    "optimes": {
        "lastCommittedOptime": {
            "ts": Timestamp(1575737656, 1),
            "t": NumberLong(2)
        },
        "lastCommittedWallTime": ISODate("2019-12-07T16:54:16.262Z"),
        "readConcernMajorityOptime": {
            "ts": Timestamp(1575737656, 1),
            "t": NumberLong(2)
        },
        "readConcernMajorityWallTime": ISODate("2019-12-07T16:54:16.262Z"),
        "appliedOptime": {
            "ts": Timestamp(1575737656, 1),
            "t": NumberLong(2)
        }
    }
}

```

Now in cluster there are only two instances. One is Primary and other is secondary.

```

rsreplica:SECONDARY> rs.status()
{
    "version": 1,
    "date": ISODate("2019-12-07T17:00:36Z"),
    "myState": 2,
    "term": NumberLong(2),
    "syncingTo": "localhost:27019",
    "syncSourceHost": "localhost:27019",
    "syncSourceId": 1,
    "infoMessage": "",
    "configVersion": 3,
    "self": true,
    "lastHeartbeatMessage": ""
},
{
    "id": 0,
    "name": "localhost:27018",
    "ip": "127.0.0.1",
    "health": 1,
    "state": 2,
    "stateStr": "SECONDARY",
    "uptime": 17510,
    "optimes": {
        "ts": Timestamp(1575738036, 1),
        "t": NumberLong(2)
    },
    "optimeDate": ISODate("2019-12-07T17:00:36Z"),
    "syncingTo": "localhost:27019",
    "syncSourceHost": "localhost:27019",
    "syncSourceId": 1,
    "infoMessage": "",
    "configVersion": 3,
    "self": true,
    "lastHeartbeatMessage": ""
},
{
    "id": 1,
    "name": "localhost:27019",
    "ip": "127.0.0.1",
    "health": 1,
    "state": 1,
    "stateStr": "PRIMARY",
    "uptime": 14430,
    "optimes": {
        "ts": Timestamp(1575738036, 1),
        "t": NumberLong(2)
    }
}

```

To know which is Primary instance from any running client use function `rs.isMaster();`