

## Exercise -----

1. Start MongoDB without authentication
2. Create Database ttn and insert some data into user collection.
3. Setup the replication with Artiber and verify that ttn database is replicated into secondary.
4. Add 2 more members to a replica set with
5. rs.add();
6. Try adding a member which already have data.

First of all i will start three instance of mongo on three different port .

Command -

`mongod --replSet taman --logpath "rs4/four.log" --dbpath rs4 --port 27021 &`

This will generate an instance of mongo on port 27021 for eg.

```
taman@taman:NodeJS $ mongod --replSet taman --logpath "one.log" --dbpath rs1 --port 27018 &
[1] 11812
taman@taman:NodeJS $ mongod --replSet taman --logpath "two.log" --dbpath rs2 --port 27019 &
[2] 11816
taman@taman:NodeJS $ mongod --replSet taman --logpath "three.log" --dbpath rs3 --port 27020 &
[3] 11900
taman@taman:NodeJS $ ps -ef | grep mongod
mongod 2611 1 0 May03 ? 00:06:13 /usr/bin/mongod --config /etc/mongod.conf
taman 11812 11765 7 15:39 pts/2 00:00:01 mongod --replSet taman --logpath one.log --dbpath rs1 --port 27018
taman 11816 11765 7 15:39 pts/2 00:00:01 mongod --replSet taman --logpath two.log --dbpath rs2 --port 27019
taman 11900 11765 7 15:39 pts/2 00:00:00 mongod --replSet taman --logpath three.log --dbpath rs3 --port 27020
taman 11956 11765 0 15:39 pts/2 00:00:00 grep --color=auto mongod
taman@taman:NodeJS $
```

Now we have to configure mongo so run mongo on port 27018 or any port which you already instanciated.

This config object will take an id for each repl set and the port address where he has to make replica or primary database. In this i want to add one instance to arbitrary so i pass a property arbitrary to true.

```

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

> config = {
...   _id:"taman",
...   members:[
...     {id: 0, host: "localhost:27018"},
...     {id: 1, host: "localhost:27019"},
...     {id: 2, host: "localhost:27020","arbiterOnly" : true}
...   ]
... }
{
  "_id" : "taman",
  "members" : [
    {
      "_id" : 0,
      "host" : "localhost:27018"
    },
    {
      "_id" : 1,
      "host" : "localhost:27019"
    },
    {
      "_id" : 2,
      "host" : "localhost:27020",
      "arbiterOnly" : true
    }
  ]
}
> rs.initiate(config);

```

Now we have done with config object now initiate the instance by passing config to a command -

`rs.initiate(config);`

```

{
  "_id" : "taman",
  "members" : [
    {
      "_id" : 0,
      "host" : "localhost:27018"
    },
    {
      "_id" : 1,
      "host" : "localhost:27019"
    },
    {
      "_id" : 2,
      "host" : "localhost:27020",
      "arbiterOnly" : true
    }
  ]
}
> rs.initiate(config);
{
  "ok" : 1,
  "$clusterTime" : {
    "clusterTime" : Timestamp(1588760168, 1),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  },
  "operationTime" : Timestamp(1588760168, 1)
}
taman:SECONDARY> |

```

We can check members which one is primary and secondary by checking status..

Command - rs.status();

Member -1

```
    "electionTimeoutMillis" : NumberLong(10000),
    "numCatchUpOps" : NumberLong(0),
    "newTermStartDate" : ISODate("2020-05-06T10:16:19.702Z"),
    "wMajorityWriteAvailabilityDate" : ISODate("2020-05-06T10:16:21.386Z")
  },
  "members" : [
    {
      "_id" : 0,
      "name" : "localhost:27018",
      "health" : 1,
      "state" : 1,
      "stateStr" : "PRIMARY",
      "uptime" : 418,
      "optime" : {
        "ts" : Timestamp(1588760181, 1),
        "t" : NumberLong(1)
      },
      "optimeDate" : ISODate("2020-05-06T10:16:21Z"),
      "syncingTo" : "",
      "syncSourceHost" : "",
      "syncSourceId" : -1,
      "infoMessage" : "could not find member to sync from",
      "electionTime" : Timestamp(1588760179, 1),
      "electionDate" : ISODate("2020-05-06T10:16:19Z"),
      "configVersion" : 1,
      "self" : true,
      "lastHeartbeatMessage" : ""
    },
  ],
}
```

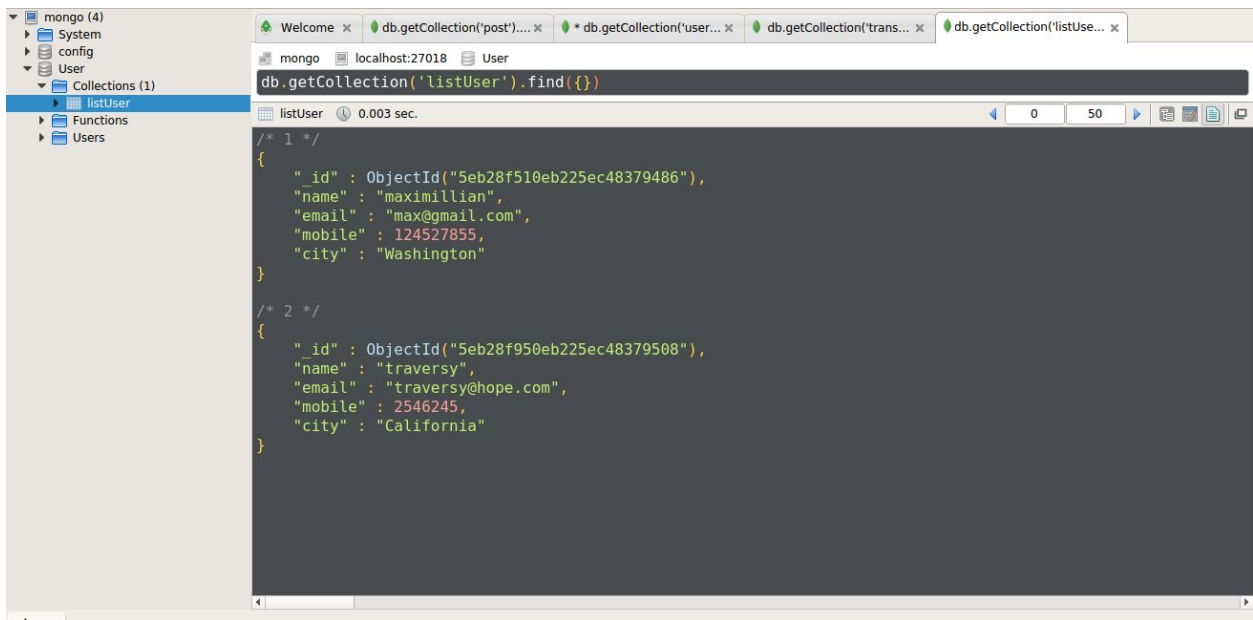
member -2

```
{
  "_id" : 1,
  "name" : "localhost:27019",
  "health" : 1,
  "state" : 2,
  "stateStr" : "SECONDARY",
  "uptime" : 24,
  "optime" : {
    "ts" : Timestamp(1588760181, 1),
    "t" : NumberLong(1)
  },
  "optimeDurable" : {
    "ts" : Timestamp(1588760181, 1),
    "t" : NumberLong(1)
  },
  "optimeDate" : ISODate("2020-05-06T10:16:21Z"),
  "optimeDurableDate" : ISODate("2020-05-06T10:16:21Z"),
  "lastHeartbeat" : ISODate("2020-05-06T10:16:31.486Z"),
  "lastHeartbeatRecv" : ISODate("2020-05-06T10:16:33.122Z"),
  "pingMs" : NumberLong(0),
  "lastHeartbeatMessage" : "",
  "syncingTo" : "localhost:27018",
  "syncSourceHost" : "localhost:27018",
  "syncSourceId" : 0,
  "infoMessage" : "",
  "configVersion" : 1
},
}
```

## Member - 3

```
{
  "id" : 2,
  "name" : "localhost:27020",
  "health" : 1,
  "state" : 7,
  "stateStr" : "ARBITER",
  "uptime" : 24,
  "lastHeartbeat" : ISODate("2020-05-06T10:16:31.487Z"),
  "lastHeartbeatRecv" : ISODate("2020-05-06T10:16:31.749Z"),
  "pingMs" : NumberLong(0),
  "lastHeartbeatMessage" : "",
  "syncingTo" : "",
  "syncSourceHost" : "",
  "syncSourceId" : -1,
  "infoMessage" : "",
  "configVersion" : 1
},
"ok" : 1,
"$clusterTime" : {
  "clusterTime" : Timestamp(1588760181, 1),
  "signature" : {
    "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAA="),
    "keyId" : NumberLong(0)
  }
},
"operationTime" : Timestamp(1588760181, 1)
```

Now i will add some data in primary database member which is running on port--27018



Now kill the process of port 27018 and now check process status..

```
taman@taman:~$ kill 11812
taman@taman:~$ ps -ef | grep mongod
mongod    2611      1   0 May03 ?        00:06:18 /usr/bin/mongod --config /etc/mongod.conf
taman     11816  11765  1 15:39 pts/2    00:00:08 mongod --replSet taman --logpath two.log --dbpath rs2 --port 27019
taman     11900  11765  0 15:39 pts/2    00:00:06 mongod --replSet taman --logpath three.log --dbpath rs3 --port 27020
taman     13285  13215  0 15:52 pts/3    00:00:00 grep --color=auto mongod
taman@taman:~$
```

Check with command `rs.status()` now member 1 is not unhealthy one.

```

"members" : [
  {
    "_id" : 0,
    "name" : "localhost:27018",
    "health" : 0,
    "state" : 8,
    "stateStr" : "(not reachable/healthy)",
    "uptime" : 0,
    "optime" : {
      "ts" : Timestamp(0, 0),
      "t" : NumberLong(-1)
    },
    "optimeDurable" : {
      "ts" : Timestamp(0, 0),
      "t" : NumberLong(-1)
    },
    "optimeDate" : ISODate("1970-01-01T00:00:00Z"),
    "optimeDurableDate" : ISODate("1970-01-01T00:00:00Z"),
    "lastHeartbeat" : ISODate("2020-05-06T10:23:15.747Z"),
    "lastHeartbeatRecv" : ISODate("2020-05-06T10:22:25.525Z"),
    "pingMs" : NumberLong(0),
    "lastHeartbeatMessage" : "Error connecting to localhost:27018 (127.0.0.1:27018) :: caused by :: Connectio
refused",
    "syncingTo" : "",
    "syncSourceHost" : "",
    "syncSourceId" : -1,
    "infoMessage" : "",
    "configVersion" : -1
  }
]

```

And member 2 will now become primary

```

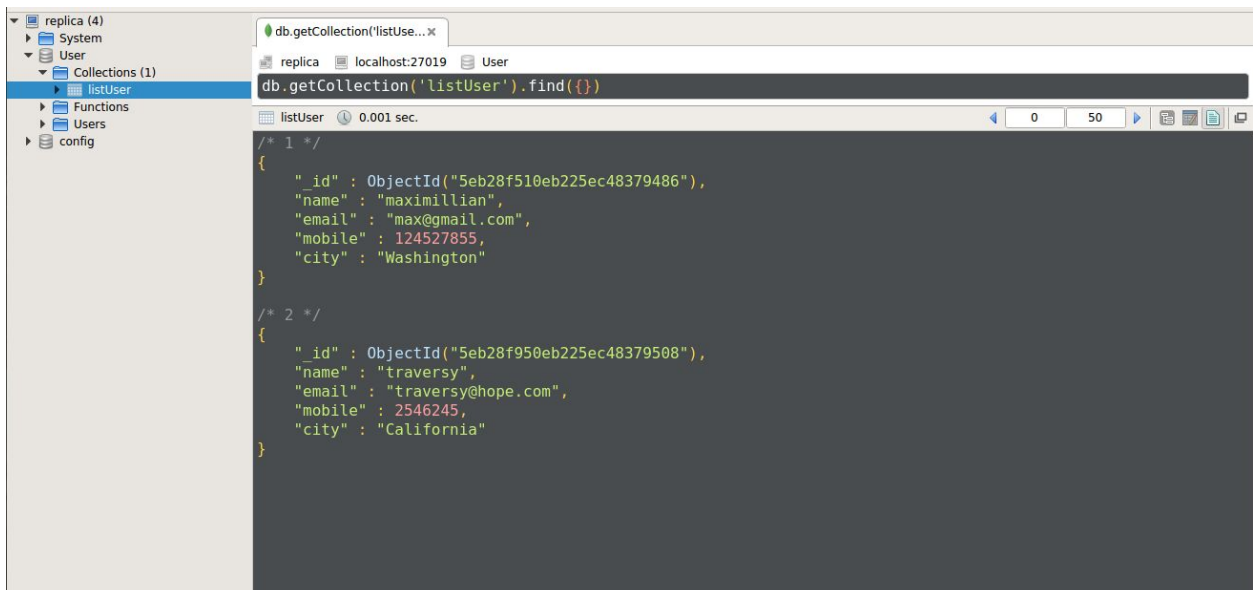
    {
      "_id" : 1,
      "name" : "localhost:27019",
      "health" : 1,
      "state" : 1,
      "stateStr" : "PRIMARY",
      "uptime" : 820,
      "optime" : {
        "ts" : Timestamp(1588760587, 1),
        "t" : NumberLong(3)
      },
      "optimeDate" : ISODate("2020-05-06T10:23:07Z"),
      "syncingTo" : "",
      "syncSourceHost" : "",
      "syncSourceId" : -1,
      "infoMessage" : "could not find member to sync from",
      "electionTime" : Timestamp(1588760557, 1),
      "electionDate" : ISODate("2020-05-06T10:22:37Z"),
      "configVersion" : 1,
      "self" : true,
      "lastHeartbeatMessage" : ""
    },
  ]
}

```

Member 3 is as usual arbitrary one which is only for voting purpose.

```
{
  "_id" : 2,
  "name" : "localhost:27020",
  "health" : 1,
  "state" : 7,
  "stateStr" : "ARBITER",
  "uptime" : 426,
  "lastHeartbeat" : ISODate("2020-05-06T10:23:15.718Z"),
  "lastHeartbeatRecv" : ISODate("2020-05-06T10:23:15.777Z"),
  "pingMs" : NumberLong(0),
  "lastHeartbeatMessage" : "",
  "syncingTo" : "",
  "syncSourceHost" : "",
  "syncSourceId" : -1,
  "infoMessage" : "",
  "configVersion" : 1
},
"ok" : 1,
"$clusterTime" : {
  "clusterTime" : Timestamp(1588760587, 1),
  "signature" : {
    "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
    "keyId" : NumberLong(0)
  }
}
```

Now we check the member 2 which is now primary database . So we already added some data in member 1 now we will confirm the data is present in second member which is now primary one or not.



The screenshot shows the MongoDB Compass interface. On the left, the 'listUser' collection is expanded under the 'User' database. The main panel displays the results of a query: `db.getCollection('listUser').find({})`. The results show two documents in the 'listUser' collection:

```
/* 1 */
{
  "_id" : ObjectId("5eb28f510eb225ec48379486"),
  "name" : "maximillian",
  "email" : "max@gmail.com",
  "mobile" : 124527855,
  "city" : "Washington"
}

/* 2 */
{
  "_id" : ObjectId("5eb28f950eb225ec48379508"),
  "name" : "traversy",
  "email" : "traversy@hope.com",
  "mobile" : 2546245,
  "city" : "California"
}
```

Now add two more instance of mongo by running two instance by same command `mongod --replSet taman --logpath "rs4/four.log" --dbpath rs4 --port 27021 &` And add two instance into primary database using



rs.add({here pass the config of the mongo instance}).

Now check `_id : 3` and `_id : 4` is also added as secondary database which has same data which is present on primary one.

```
{
  "_id" : 3,
  "name" : "localhost:27021",
  "health" : 1,
  "state" : 2,
  "stateStr" : "SECONDARY",
  "uptime" : 29,
  "optime" : {
    "ts" : Timestamp(1588761177, 1),
    "t" : NumberLong(3)
  },
  "optimeDurable" : {
    "ts" : Timestamp(1588761177, 1),
    "t" : NumberLong(3)
  },
  "optimeDate" : ISODate("2020-05-06T10:32:57Z"),
  "optimeDurableDate" : ISODate("2020-05-06T10:32:57Z"),
  "lastHeartbeat" : ISODate("2020-05-06T10:33:04.624Z"),
  "lastHeartbeatRecv" : ISODate("2020-05-06T10:33:05.132Z"),
  "pingMs" : NumberLong(0),
  "lastHeartbeatMessage" : "",
  "syncingTo" : "localhost:27019",
  "syncSourceHost" : "localhost:27019",
  "syncSourceId" : 1,
  "infoMessage" : "",
  "configVersion" : 3
},
```

```
{
  "_id" : 4,
  "name" : "localhost:27022",
  "health" : 1,
  "state" : 2,
  "stateStr" : "SECONDARY",
  "uptime" : 18,
  "optime" : {
    "ts" : Timestamp(1588761177, 1),
    "t" : NumberLong(3)
  },
  "optimeDurable" : {
    "ts" : Timestamp(1588761177, 1),
    "t" : NumberLong(3)
  },
  "optimeDate" : ISODate("2020-05-06T10:32:57Z"),
  "optimeDurableDate" : ISODate("2020-05-06T10:32:57Z"),
  "lastHeartbeat" : ISODate("2020-05-06T10:33:04.624Z"),
  "lastHeartbeatRecv" : ISODate("2020-05-06T10:33:04.343Z"),
  "pingMs" : NumberLong(0),
  "lastHeartbeatMessage" : "",
  "syncingTo" : "localhost:27021",
  "syncSourceHost" : "localhost:27021",
  "syncSourceId" : 3,
  "infoMessage" : "",
  "configVersion" : 3
}
```

Now i trying to add new member which has some other data and i am adding it in current primary mongo instance.

This will throw an error..

```
        "clusterTime" : Timestamp(1588761177, 1),
        "signature" : {
          "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAA="),
          "keyId" : NumberLong(0)
        }
      },
      "operationTime" : Timestamp(1588761177, 1)
    }
  }
}
taman:PRIMARY> rs.add({_id: 5, host: "localhost:27019"})
{
  "operationTime" : Timestamp(1588761267, 1),
  "ok" : 0,
  "errmsg" : "Found two member configurations with same host field, members.1.host == members.5.host == localhost:27019",
  "code" : 103,
  "codeName" : "NewReplicaSetConfigurationIncompatible",
  "$clusterTime" : {
    "clusterTime" : Timestamp(1588761267, 1),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
taman:PRIMARY> |
```