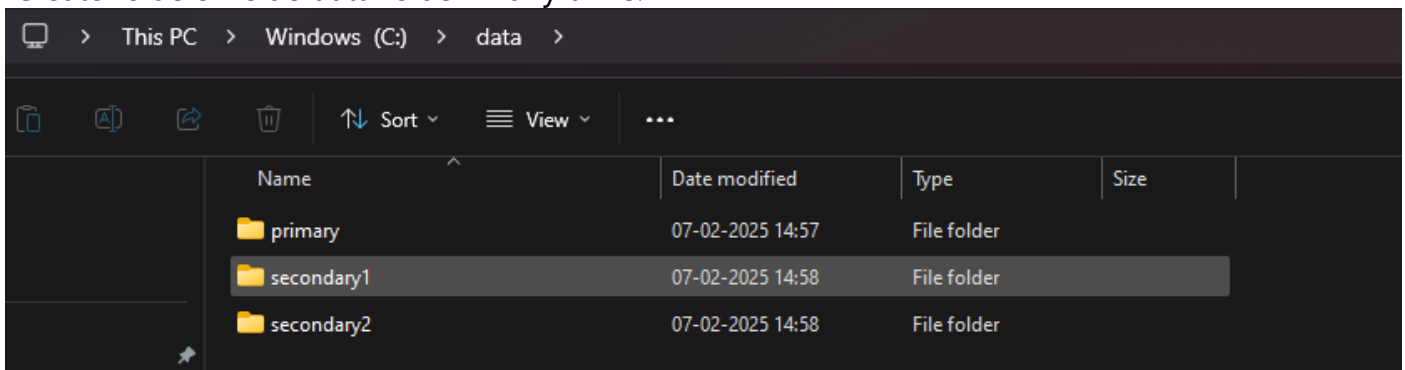


## PRACTICAL 5 REPLICATION USING MONGODB

You are a database administrator for a company, and you need to set up a MongoDB replica set to ensure high availability and data redundancy. Perform the following tasks:

- Initialize a replica set with three nodes on different ports (27017, 27018, 27019).
- Check the status of the replica set.
- Add a new secondary node to the existing replica set.
- Simulate a failover scenario by stepping down the primary and observing the election of a new primary.
- Check replication status and read from a secondary node using readPreference

Create folders inside data folder in any drive.



Open Windows PowerShell ISE and execute the below commands.

```
mongod --port=2717 --dbpath="C:\data\primary" --replSet="text-replica-set"
```

```
PS C:\Users\Admin> mongod --port=2717 --dbpath="C:\data\primary" --replSet="test-replica-set"
{"t":{"$date":"2025-02-07T15:22:00.601+05:30"},"s":"I", "c":"CONTROL", "id":23285, "ctx":"thread1", "msg":"A
{"t":{"$date":"2025-02-07T15:22:00.602+05:30"},"s":"I", "c":"NETWORK", "id":4915701, "ctx":"thread1", "msg":"I
reVersion":17}, "incomingInternalClient":{"minWireVersion":0, "maxWireVersion":17}, "outgoing":{"minWireVersion":6
{"t":{"$date":"2025-02-07T15:22:00.603+05:30"},"s":"I", "c":"NETWORK", "id":4648602, "ctx":"thread1", "msg":"I
{"t":{"$date":"2025-02-07T15:22:00.604+05:30"},"s":"I", "c":"REPL", "id":5123008, "ctx":"thread1", "msg":"S
ace":"config.tenantMigrationDonors"}}
{"t":{"$date":"2025-02-07T15:22:00.604+05:30"},"s":"I", "c":"REPL", "id":5123008, "ctx":"thread1", "msg":"S
mespace":"config.tenantMigrationRecipients"}}
{"t":{"$date":"2025-02-07T15:22:00.604+05:30"},"s":"I", "c":"REPL", "id":5123008, "ctx":"thread1", "msg":"S
"config.tenantSplitDonors"}}
{"t":{"$date":"2025-02-07T15:22:00.607+05:30"},"s":"I", "c":"CONTROL", "id":5945603, "ctx":"thread1", "msg":"M
{"t":{"$date":"2025-02-07T15:22:00.608+05:30"},"s":"I", "c":"CONTROL", "id":4615611, "ctx":"initandlisten", "m
64-bit", "host":"MUM2024COM07"}}
{"t":{"$date":"2025-02-07T15:22:00.608+05:30"},"s":"I", "c":"CONTROL", "id":23398, "ctx":"initandlisten", "m
R2"}}
{"t":{"$date":"2025-02-07T15:22:00.608+05:30"},"s":"I", "c":"CONTROL", "id":23403, "ctx":"initandlisten", "m
```

Open new shell and type the command

```
mongod --port=2727 --dbpath="C:\data\secondary1" --replSet="text-replica-set"
```

```
PS C:\Users\Admin> mongod --port=2727 --dbpath="C:\data\secondary1" --replSet="test-replica-set"
{"t":{"$date":"2025-02-07T15:25:21.987+05:30"},"s":"I", "c":"NETWORK", "id":4915701, "ctx":"thread1", "
d wire specification", "attr":{"spec":{"incomingExternalClient":{"minWireVersion":0, "maxWireVersion":17},
lClient":{"minWireVersion":0, "maxWireVersion":17}, "outgoing":{"minWireVersion":6, "maxWireVersion":17}, "i
:true}}}}
{"t":{"$date":"2025-02-07T15:25:22.813+05:30"},"s":"I", "c":"CONTROL", "id":23285, "ctx":"thread1", "
lly disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
{"t":{"$date":"2025-02-07T15:25:22.815+05:30"},"s":"I", "c":"NETWORK", "id":4648602, "ctx":"thread1", "
CP FastOpen in use."}
```

Open new shell and type the command

```
mongod --port=2737 --dbpath="C:\data\secondary2" --replSet="text-replica-set"
```

```
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Admin> mongod --port=2737 --dbpath="C:\data\secondary2" --replSet="test-replica-set"
{"t":{"$date":"2025-02-07T15:26:58.423+05:30"},"s":"I", "c":"CONTROL", "id":23285, "ctx":"-", "msg":"Automatically di
sabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'"}
{"t":{"$date":"2025-02-07T15:26:59.286+05:30"},"s":"I", "c":"NETWORK", "id":4915701, "ctx":"-", "msg":"Initialized wire
specification", "attr":{"spec":{"incomingExternalClient":{"minWireVersion":0,"maxWireVersion":17},"incomingInternalClien
t":{"minWireVersion":0,"maxWireVersion":17},"outgoing":{"minWireVersion":6,"maxWireVersion":17},"isInternalClient":true}
}}
{"t":{"$date":"2025-02-07T15:26:59.290+05:30"},"s":"I", "c":"NETWORK", "id":4648602, "ctx":"thread1", "msg":"Implicit T
CP FastOpen in use."}
{"t":{"$date":"2025-02-07T15:26:59.291+05:30"},"s":"I", "c":"REPL", "id":5123008, "ctx":"thread1", "msg":"Successful
ly registered PrimaryOnlyService" "attr":{"service":"TenantMigrationDonorService" "namespace":"config.tenantMigrationDon
```

## Initialization of primary server and adding secondary members to it

```
mongosh --host="localhost:2717"
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Admin> mongosh host="localhost:2717"
Current Mongosh Log ID: 67a5d99334813c8483ffb203
Connecting to:      mongodb://127.0.0.1:27017/host%3Dlocalhost%3A2717?directConnection=true&serv
Using MongoDB:      6.0.13
Using Mongosh:      2.1.4
mongosh 2.3.8 is available for download: https://www.mongodb.com/try/download/shell

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

-----
The server generated these startup warnings when booting
2025-01-18T10:42:11.533+05:30: Access control is not enabled for the database. Read and write acc
-----

host=localhost:2717> |
```

```
rs.initiate();
```

```
test> rs.initiate()
{
  info2: 'no configuration specified. Using a default confi
me: 'localhost:2717',
  ok: 1
}
test-replica-set [direct: other] test> |
```

## To add configurations to primary server

```
rs.add({host:"localhost:2727"});
```

```
test-replica-set [direct: other] test> rs.add({host:"localhost:2727"})
{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1738922805, i: 1 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1738922805, i: 1 })
}
```

rs.add({host:"localhost:2737"});

```
test-replica-set [direct: primary] test> rs.add({host:"localhost:2737"})
{
  ok: 1,
  '$clusterTime': {
    clusterTime: Timestamp({ t: 1738922839, i: 1 }),
    signature: {
      hash: Binary.createFromBase64('AAAAAAAAAAAAAAAAAAAAAAAAAAAA=', 0),
      keyId: Long('0')
    }
  },
  operationTime: Timestamp({ t: 1738922839, i: 1 })
}
test-replica-set [direct: primary] test> |
```

rs.status();

```
test-replica-set [direct: primary] test> rs.status()
{
  set: 'test-replica-set',
  date: ISODate('2025-02-07T10:09:29.871Z'),
  myState: 1,
  term: Long('1'),
  syncSourceHost: '',
  syncSourceId: -1,
  heartbeatIntervalMillis: Long('2000'),
  majorityVoteCount: 2,
}
```

mongosh --host="localhost:2727"

```
PS C:\Users\Admin> mongosh --host="localhost:2727"
Current Mongosh Log ID: 67a5dc739b1f7c03753892c3
Connecting to:      mongodb://localhost:2727/?directConnection=1
1.4
Using MongoDB:      6.0.13
Using Mongosh:      2.1.4
mongosh 2.3.9 is available for download: https://www.mongodb.com/

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

-----
The server generated these startup warnings when booting
2025-02-07T15:25:22.852+05:30: Access control is not enabled f
figuration is unrestricted
2025-02-07T15:25:22.852+05:30: This server is bound to localho
rver. Start the server with --bind_ip <address> to specify which
bind_ip_all to bind to all interfaces. If this behavior is desire
this warning
-----

test-replica-set [direct: secondary] test> |
```

mongosh --host="localhost:2737"

```
PS C:\Users\Admin> mongosh --host="localhost:2737"
Current Mongosh Log ID: 67a5dcaccc29d0e1cac27edb
Connecting to:      mongodb://localhost:2737/?directConn
1.4
Using MongoDB:      6.0.13
Using Mongosh:      2.1.4
mongosh 2.3.9 is available for download: https://www.mongodb.com
For mongosh info see: https://docs.mongodb.com/mongodb-shell

-----
The server generated these startup warnings when booting
2025-02-07T15:26:59.324+05:30: Access control is not enabled
configuration is unrestricted
2025-02-07T15:26:59.324+05:30: This server is bound to localhost
server. Start the server with --bind_ip <address> to specify which
bind_ip_all to bind to all interfaces. If this behavior is desired,
this warning
-----

test-replica-set [direct: secondary] test> |
```

Now, go to primary instance and type the following command  
show dbs;

```
test-replica-set [direct: primary] test> show dbs
admin      80.00 KiB
config     176.00 KiB
local      404.00 KiB
test-replica-set [direct: primary] test> |
```

Go to secondary1 and type the same command  
show dbs;

```
test-replica-set [direct: secondary] test> show dbs
admin      80.00 KiB
config     220.00 KiB
local      404.00 KiB
test-replica-set [direct: secondary] test> |
```

Go to secondary2 instance and type the same command  
show dbs;

```
test-replica-set [direct: secondary] test> show dbs
admin      80.00 KiB
config     220.00 KiB
local      404.00 KiB
test-replica-set [direct: secondary] test> |
```

## Adding a database in primary server and creating a table with a record and using find() use practice

db.books.insert({name:"think like a monk"})

```
Switched to db Shweta
test-replica-set [direct: primary] Shweta> db.books.insert({name:"think like a monk"})
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId('67a5df2269a68aef79a7cc5d') }
}
```

```
test-replica-set [direct: primary] Shweta> db.books.insertMany([
... { name:"Do not belive everything you think",
...   price:250},
... { name:"Atomic Habbit",
...   price:200 },
... { name:"Mindset",
...   price=150 }])
... ;
```

```
test-replica-set [direct: primary] Shweta> db.books.find()
[
  {
    _id: ObjectId('67a5df2269a68aef79a7cc5d'),
    name: 'think like a monk'
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc5e'),
    name: 'Do not belive everything you think',
    price: 250
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc5f'),
    name: 'Atomic Habbit',
    price: 200
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc60'),
    name: 'Mindset',
    price: 150
  }
]
test-replica-set [direct: primary] Shweta> |
```

Now, updating the records..

```
test-replica-set [direct: primary] Shweta> db.books.updateOne({ name:"Mindset"}, { $set: { price:200}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
test-replica-set [direct: primary] Shweta> db.books.find()
[
  {
    _id: ObjectId('67a5df2269a68aef79a7cc5d'),
    name: 'think like a monk'
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc5e'),
    name: 'Do not belive everything you think',
    price: 250
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc5f'),
    name: 'Atomic Habbit',
    price: 200
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc60'),
    name: 'Mindset',
    price: 200
  }
]
```

Now , deleting the records..

```
test-replica-set [direct: primary] Shweta> db.books.deleteOne({ name:"mindset"})
{ acknowledged: true, deletedCount: 0 }
test-replica-set [direct: primary] Shweta> |
```

Check if the records are visible in secondary1 and secondary2

```
test-replica-set [direct: secondary] Shweta> db.getMongo().setReadPref("secondaryPreferred")
```

```
test-replica-set [direct: secondary] Shweta> db.books.find();
[
  {
    _id: ObjectId('67a5df2269a68aef79a7cc5d'),
    name: 'think like a monk'
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc5f'),
    name: 'Atomic Habbit',
    price: 200
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc5e'),
    name: 'Do not belive everything you think',
    price: 250
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc60'),
    name: 'Mindset',
    price: 200
  }
]
```

```
test-replica-set [direct: secondary] Shweta> db.books.find();
[
  {
    _id: ObjectId('67a5df2269a68aef79a7cc5d'),
    name: 'think like a monk'
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc5f'),
    name: 'Atomic Habbit',
    price: 200
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc5e'),
    name: 'Do not belive everything you think',
    price: 250
  },
  {
    _id: ObjectId('67a5e0f869a68aef79a7cc60'),
    name: 'Mindset',
    price: 200
  }
]
```