**MongoDB Replication**

A *replica set* in MongoDB is a group of [mongod](https://www.mongodb.com/docs/manual/reference/program/mongod/#mongodb-binary-bin.mongod) processes that maintain the same data set. Replica sets provide redundancy and [high availability](https://www.mongodb.com/docs/manual/reference/glossary/#std-term-high-availability).  
  
In simple terms, MongoDB replication is the process of creating a copy of the same data set in more than one MongoDB server. This can be achieved by using a Replica Set. A replica set is a group of MongoDB instances that maintain the same data set and pertain to any mongod process.

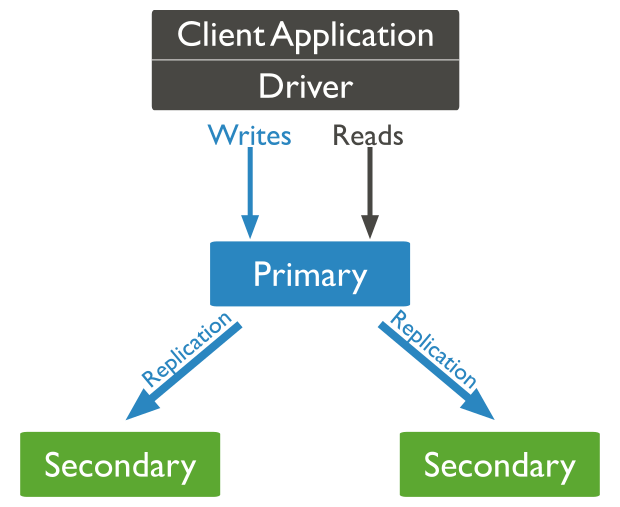
Maintaining multiple MongoDB servers with the same data provides distributed access to the data while increasing the fault tolerance of the database by providing backups.

**How MongoDB replication works ?**

MongoDB handles replication through a Replica Set, which consists of multiple MongoDB nodes that are grouped together as a unit.

A Replica Set requires a minimum of three MongoDB nodes:

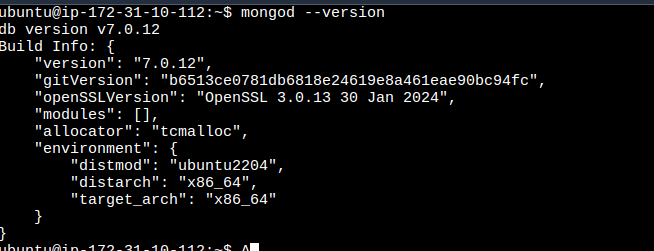
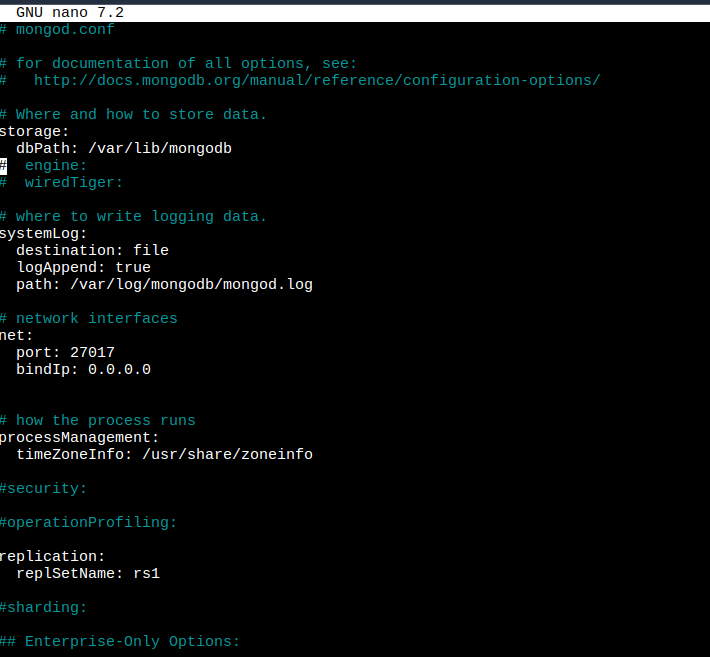
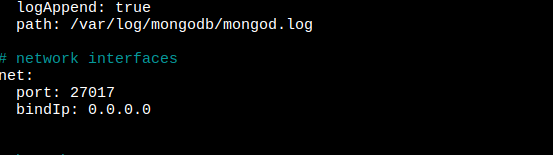
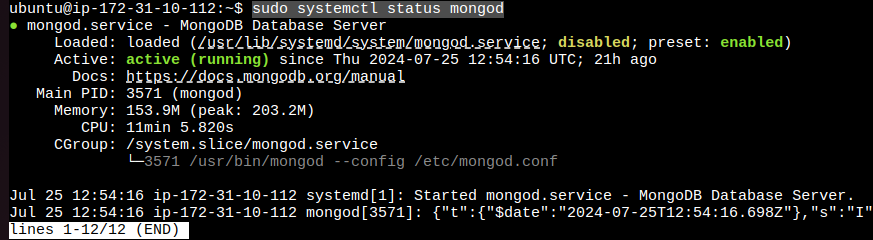
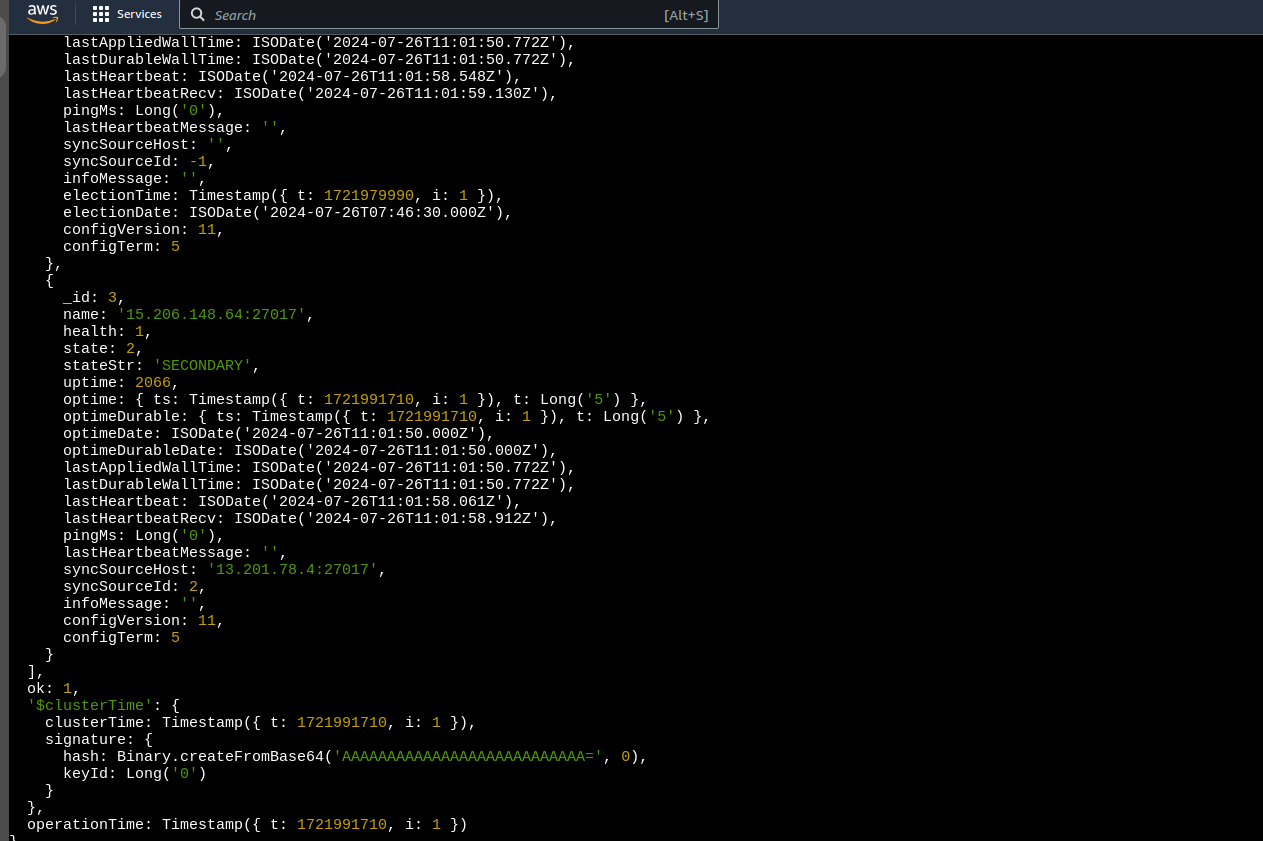
* One of the nodes will be considered the primary node that receives all the write operations.
* The others are considered secondary nodes. These secondary nodes will replicate the data from the primary node.



*Basic replication methodology*

While the primary node is the only instance that accepts write operations, any other node within a replica set can accept read operations. These can be configured through a supported MongoDB client.

**SETUP on AWS EC2**

1. Create EC2 instance (need to have minimum 3 instance as min number of db/nodes need is 3)
   1. Install monogdb in all the instances.
   2. Go to mongo.conf file & updateconfiguration.  
       **sudo nano /etc/mongod.conf**monogd.conf file will look like above sscreenshot
   3. Update net: bindIp to 0.0.0.0
   4. Uncomment replication and set replSetName:rs1, while adding replSetName or any child key, do remember to add 2 spaces (no less/more) otherwise it will give error while starting monog services.  
      **replSetName:rs1 -> here rs1 is the name of replica set (it can be anything)**
   5. After saving monogd.conf file changes, restart mongo server using   
      **sudo systemctl restart mongod**
   6. Check status of monogd server using following command  
      **sudo systemctl status mongod **
   7. **Perform all the above steps in all the instances which will be part of db replication. The replSetName should be same for all monogdb instances.**
   8. Open monogshell in 1st instance (ie 1st database which will be the primary database/node)   
      Execute **mongosh** to open mongo shell.
   9. Execute following sets of command in shell (in primary node only)  
      **rs.initiate();  
      rs.add(‘public-ip-2nd-insatnce:port’)** //Eg: res.add(‘1.1.1.1:27017’)   
      **rs.add(‘public-ip-3nd-insatnce:port’)** //Eg:res.add(‘1.2.2.1:27017’)
   10. Execute **rs.status()** to check the status of replicaSet.  
         
       Will display the list of members with thor status and stateStr = PRIMIRY || SECONDARY
   11. Exit the terminal and reopen monogshell you will get to see .  
         
       *In primary instance shell*   
         
         
       *In secondary instance shell (for both 2nd and 3rd instance) both will be considered as secondary*  
       

The primary database will handle both read and write operations, while secondary databases will be used exclusively for read operations.  
Any operation /add/delete/update which will be performed on the Primary database will be replicated in both secondary databases, If the PRIMARYdatabase fails due to any reason(server error, database hacking etc) , the 1st secondary database will be switched as a primary , and so on. As soon as the PRIMARYdatabase/ server recovers , it will become as PRIMARY (same as its initial state) and then temporary PRIMARY(ie 1st secondary) will switch to its initial state, i,e SECONDARY.  
  
You cannot directly perform any write operation on a SECONDARY db  
  
To TEST if replication is working properly, you can perform the CRUD in PRIMARY database and it will get reflected in SECONDARY.  
  
**MonogDB url to be used in NodeJs   
 MONGO\_URL = "mongodb://hostIP1:27017,hostIP2:27017,hostIP3:27017/transactionDb?&replicaSet=rs1"**