03 de Diciembre de 2020

**ASIGNATURA:** Bases de Datos Avanzadas

**Edwar Alonso Rojas Blanco**
edrojasb@unal.edu.co

# Cluster MongoDB

**https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu-tarball/**

## Instalación de Mongo (Opción aconsejable, v4)

**1.** Instalar
https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu/

Para: **Ubuntu 20**

```
wget -qO - https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add -
echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.4.list
sudo apt-get update
sudo apt-get install -y mongodb-org
```

2. Servicio mongo

```
sudo systemctl start mongod
sudo systemctl daemon-reload
sudo systemctl status mongod

service mongod status
```

3. Iniciar el servisio con el sistema

Opcional

```
sudo systemctl enable mongod (enable on reboot)
sudo systemctl disable mongod
```

4. Log

```
sudo tail /var/log/mongodb/mongod.log
```

5. Config

```
sudo cat /etc/mongod.conf
```

6. Probar instlaación

```
mongo
```

# Instalar desde Binario

1. Descargue MongoDB: https://fastdl.mongodb.org/linux/mongodb-linux-x86_64-3.2.9.tgz

2. Descomprima y cambie al directorio creado:

```
$ tar xfz mongodb-linux-x86_64-3.2.9.tgz
$ cd mongodb-linux-x86_64-3.2.9
```

3. Crear el directorio de datos:

```
$ mkdir ./data
```

4. Inicie el servidor de MongoDB:

```
$ ./bin/mongod --dbpath ./data
```

5. Inicie el interprete de comandos (Mongo shell):

```
$ ./bin/mongo
```

# Sharding

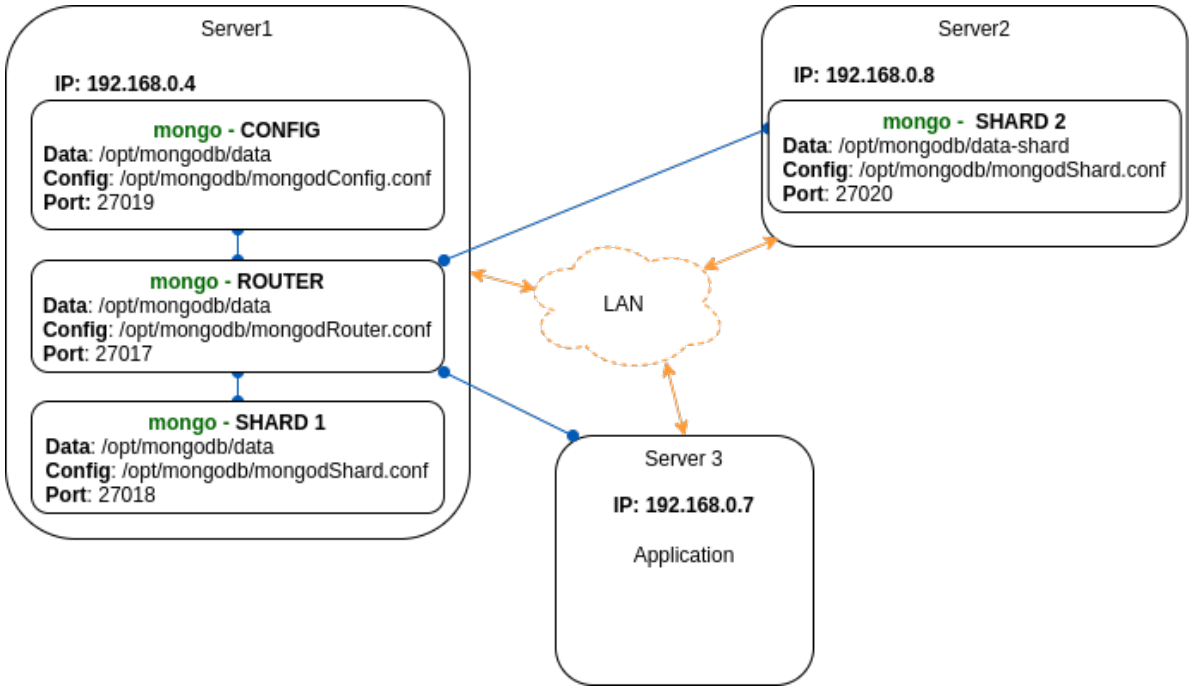
**Configuración del Clúster**

**https://docs.mongodb.com/manual/sharding/**

**https://www.bmc.com/blogs/mongodb-sharding-explained/**

**(Se recomienda la misma versión de mongo en todos los servidores)**

**Arquitectura**



## Configurando "Config server" (Importante ejecutar comandos como root)

```
server1# cd /opt/mongodb
server1# mkdir –pv  data–config/configdb/
server1# mkdir –pv  data–config/logs
server1# touch data–config/logs/configsvr.log
server1# ls –alRv ./
```

```
server1# vim mongodConfig.conf
```

```
# mongod.conf

# for documentation of all options, see:
#   http://docs.mongodb.org/manual/reference/configuration–options/

# Where and how to store data.
storage:
  dbPath: /opt/mongodb/data–config/configdb
  journal:
    enabled: true
#  engine:
#  mmapv1:
#  wiredTiger:

# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path: /opt/mongodb/data–config/logs/configsvr.log

# network interfaces
net:
  port: 27019
  bindIp: 192.168.0.4

sharding:
  clusterRole: configsvr

replication:
  replSetName: ConfigReplSet


# how the process runs
#processManagement:
#  timeZoneInfo: /usr/share/zoneinfo

#security:

#operationProfiling:

#replication:

#sharding:

## Enterprise–Only Options:

#auditLog:

#snmp:
```

```
server1# mongod --config /opt/mongodb/mongodConfig.conf&
```

```
server1# tail /opt/mongodb/data-config/logs/configsvr.log
```

```
server1# mongo  192.168.0.4:27019
```

```
> rs.initiate()
> rs.status()
```

## Configurando "Router"

```
server1# cd /opt/mongodb
```

```
server1# mkdir data-router
```

```
server1# mkdir -pv data-router/logs
server1# touch data-router/logs/mongorouter.log
server1# ls -alRv data-router/
```

```
server1# vim mongoRouter.conf
```

```
systemLog:
  destination: file
  logAppend: true
  path: /opt/mongodb/data-router/logs/queryrouter.log

net:
  port: 27017
  bindIp: 192.168.0.4

sharding:
  configDB: ConfigReplSet/192.168.0.4:27019
```

```
server1# mongos --config /opt/mongodb/mongoRouter.conf&
```

```
server1# mongo 192.168.0.4:27017
```

```
mongos> sh.status()
```

```
mongos> sh.getBalancerState()
```

## Configurando "Shard 1" (lo mismo en todos los shard: shard1, shard2, …. shard_n)

```
server1# cd /opt/mongodb
```

```
server1# mkdir data-shard
```

```
server1# mkdir -pv  data-shard/sharddb/
server1# mkdir -pv  data-shard/logs
server1# touch data-shard/logs/shard.log
server1# ls -alRv data-shard/
```

```
server1# vim mongodShard.conf
```

```
storage:
  dbPath: /opt/mongodb/data-shard/sharddb
  journal:
    enabled: true

systemLog:
  destination: file
  logAppend: true
  path: /opt/mongodb/data-shard/logs/shard.log

net:
  port: 27018
  bindIp: 192.168.0.4

sharding:
  clusterRole: shardsvr

replication:
  replSetName: "rs1"
```

```
server1# mongod --config mongodShard.conf&
```

```
server1# tail -100 data-shard/logs/shard.log
```

```
server1# mongo 192.168.0.4:27018
```

```
> rs.initiate()
> rs.status()
```

Chequeando: config, router y shard

```
server1# ps -A | grep mongo
```

## Agregando "Shard "

En el **router** 192.168.0.4:**27017**

```
server1# mongo  192.168.0.4:27017
```

```
mongos> sh.addShard( "rs1/192.168.0.4:27018")
mongos> sh.addShard( "ShardReplSet/192.168.0.8:27020")
```

## Base de datos Fragmentada

```
mongos> use persons
mongos> sh.enableSharding("persons")
```

```
mongos> sh.status()
```

## Creating the sharding dataset

```
mongos> db.createCollection("personscollection")
```

Create the index and add a record. Create the index with personid as the field in descending order.

```
mongos> db.personscollection.createIndex({personid: –1})
```

Add a personid with value 10001.

```
mongos> db.personscollection.insertOne({personid: 10001})
```

Enabling sharding for the “personscollection”

```
mongos> db.personscollection.ensureIndex({personid : "hashed"})
```

If the index is used correctly, we can enable Sharding with “personid” as the shard key.

```
mongos> sh.shardCollection("persons.personscollection", {personid : "hashed"})
```

Verify if sharding is working as intended. Use the getShardDistribution() command to verify the status of the sharding operation.

```
mongos> db.personscollection.getShardDistribution()
```