Actividad 3 - Conceptos y comandos básicos de la replicación en bases de datos NoSQL

Jeisson Andrés López Abril

Héctor Andrés Leyton Muñoz

Facultad de Ingeniería, Corporación Universitaria Iberoamericana

Ingeniería de software virtual

Docente Adán Beltrán

01 de Octubre de 2021

Desarrollo

- Requerimientos no funcionales relacionados con disponibilidad y redundancia requeridos

Requerimientos de disponibilidad

- El acceso a la BD debe brindar una disponibilidad del 99%
- Se debe garantizar el registro de la información de operaciones CRUD en la base de datos a través de la replicación aun cuando alguno de los nodos se baje de forma manual.
- Cada petición debe tener una respuesta indicando si fue exitosa

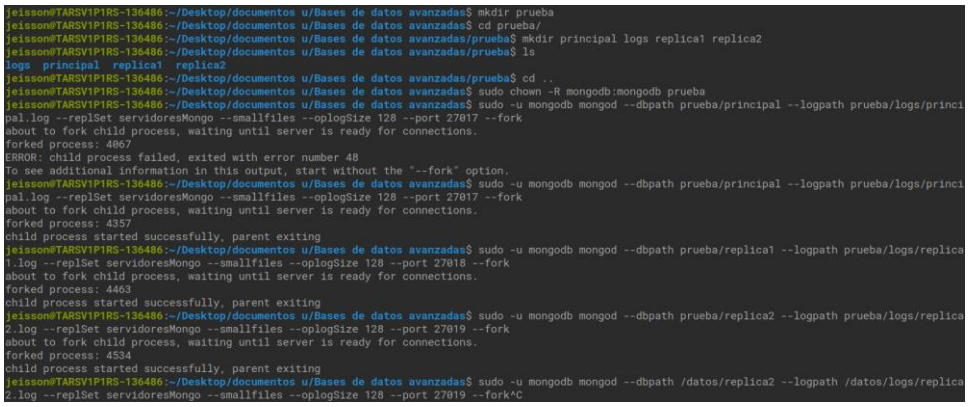
Requerimientos de redundancia

- Garantizar copias idénticas de los datos en varios servidores, de forma que, si uno de ellos falla, la información seguirá estando disponible.
- Las escrituras siempre deberán hacerse en el servidor o nodo primario.
- La replicación debe estar en la capacidad de asignar un nuevo nodo principal en caso de pérdida de este.

Requerimientos de ambiente

- Garantizar la compatibilidad bajo algún sistema operativo Linux.
- Se deberá utilizar mongobd en versiones 3 o superiores.

- Scripts de replicación



```
jeisson@TARSV1P1RS-136486:~/Desktop/documentos u/Bases de datos avanzadas$ mongo --port 27017
MongoDB shell version v3.6.3
connecting to: mongodb://127.0.0.1:27017/
MongoDB server version: 3.6.3
Server has startup warnings:
2021-09-29T23:04:29.759-0500 I STORAGE  [initandlisten]
2021-09-29T23:04:29.759-0500 I STORAGE  [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
2021-09-29T23:04:29.759-0500 I STORAGE  [initandlisten] **          See http://dochub.mongodb.org/core/prodnotes-filesystem
2021-09-29T23:04:33.857-0500 I CONTROL  [initandlisten]
2021-09-29T23:04:33.857-0500 I CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the database.
2021-09-29T23:04:33.857-0500 I CONTROL  [initandlisten] **          Read and write access to data and configuration is unrestricted.
2021-09-29T23:04:33.857-0500 I CONTROL  [initandlisten]
2021-09-29T23:04:33.857-0500 I CONTROL  [initandlisten] ** WARNING: This server is bound to localhost.
2021-09-29T23:04:33.857-0500 I CONTROL  [initandlisten] **          Remote systems will be unable to connect to this server.
2021-09-29T23:04:33.857-0500 I CONTROL  [initandlisten] **          Start the server with --bind_ip <address> to specify which IP
2021-09-29T23:04:33.857-0500 I CONTROL  [initandlisten] **          addresses it should serve responses from, or with --bind_ip_all to
2021-09-29T23:04:33.857-0500 I CONTROL  [initandlisten] **          bind to all interfaces. If this behavior is desired, start the
2021-09-29T23:04:33.857-0500 I CONTROL  [initandlisten] **          server with --bind_ip 127.0.0.1 to disable this warning.
2021-09-29T23:04:33.857-0500 I CONTROL  [initandlisten]
```

```
> cfg = {"_id":"servidoresMongo", "members":[
... {"_id":0, "host":"localhost:27017"},
... {"_id":1, "host":"localhost:27018"},
... {"_id":2, "host":"localhost:27019"}
... ]
... }
{
        "_id" : "servidoresMongo",
        "members" : [
                {
                        "_id" : 0,
                        "host" : "localhost:27017"
                },
                {
                        "_id" : 1,
                        "host" : "localhost:27018"
                },
                {
                        "_id" : 2,
                        "host" : "localhost:27019"
                }
        ]
}
> rs.initiate(cfg)
{
        "ok" : 1,
        "operationTime" : Timestamp(1632974799, 1),
        "$clusterTime" : {
                "clusterTime" : Timestamp(1632974799, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
}
servidoresMongo:SECONDARY>
servidoresMongo:PRIMARY>
servidoresMongo:PRIMARY>
servidoresMongo:PRIMARY>
servidoresMongo:PRIMARY>
servidoresMongo:PRIMARY>
servidoresMongo:PRIMARY>
servidoresMongo:PRIMARY>
servidoresMongo:PRIMARY> ^C
bye
```

```
servidoresMongo:PRIMARY> rs.status()
{
        "set" : "servidoresMongo",
        "date" : ISODate("2021-09-30T04:07:57.713Z"),
        "myState" : 1,
        "term" : NumberLong(1),
        "heartbeatIntervalMillis" : NumberLong(2000),
        "optimes" : {
                "lastCommittedOpTime" : {
                        "ts" : Timestamp(1632974875, 1),
                        "t" : NumberLong(1)
                },
                "readConcernMajorityOpTime" : {
                        "ts" : Timestamp(1632974875, 1),
                        "t" : NumberLong(1)
                },
                "appliedOpTime" : {
                        "ts" : Timestamp(1632974875, 1),
                        "t" : NumberLong(1)
                },
                "durableOpTime" : {
                        "ts" : Timestamp(1632974875, 1),
                        "t" : NumberLong(1)
                }
        },
        "members" : [
                {
                        "_id" : 0,
                        "name" : "localhost:27017",
                        "health" : 1,
                        "state" : 1,
                        "stateStr" : "PRIMARY",
                        "uptime" : 208,
                        "optime" : {
                                "ts" : Timestamp(1632974875, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDate" : ISODate("2021-09-30T04:07:55Z"),
                        "infoMessage" : "could not find member to sync from",
                        "electionTime" : Timestamp(1632974814, 1),
                        "electionDate" : ISODate("2021-09-30T04:06:54Z"),
                        "configVersion" : 1,
                        "self" : true
                },
                {
                        "_id" : 1,
                        "name" : "localhost:27018",
                        "health" : 1,
                        "state" : 2,
                        "stateStr" : "SECONDARY",
                        "uptime" : 77,
                        "optime" : {
                                "ts" : Timestamp(1632974875, 1),
                                "t" : NumberLong(1)
                        },
                        "optimeDurable" : {
                                "ts" : Timestamp(1632974875, 1),
                                "t" : NumberLong(1)
                        },
```

Según lo evidenciado en cada imagen, el proceso se trabajó en un mongo instalado en Linux (ubuntu), lo que requería realizar lo siguiente

mkdir /datos
mkdir /datos/principal
mkdir /datos/replica1
mkdir /datos/replica2
mkdir /datos/logs
sudo chown -R mongodb:mongodb /datos

Para el servidor principal quedaría de la siguiente manera.

sudo -u mongodb mongod --dbpath /datos/principal --logpath /datos/logs/principal.log --replSet servidoresMongo --smallfiles --oplogSize 128 --port 27017 --fork

Replica 1:

sudo -u mongodb mongod --dbpath /datos/replica1 --logpath /datos/logs/replica1.log --replSet servidoresMongo --smallfiles --oplogSize 128 --port 27018 --fork

Replica 2:

sudo -u mongodb mongod --dbpath /datos/replica2 --logpath /datos/logs/replica2.log --replSet servidoresMongo --smallfiles --oplogSize 128 --port 27019 --fork

mongo --port 27017

```
cfg = {"_id":"servidoresMongo", "members":[
{"_id":0, "host":"localhost:27017"},
{"_id":1, "host":"localhost:27018"},
{"_id":2, "host":"localhost:27019"}
]
}
```

rs.initiate(cfg)

rs.status()

Las imágenes finales muestran los tres nodos, principal, replica 1 y replica 2

- Video

https://laiberocol-my.sharepoint.com/:v:/g/personal/jlopezab_ibero_edu_co/EadFabAd_RpHp1zTTWYAx38B_k7YMAkl8A7og3QhEPPw8A?e=9TQDhW