



Actividad 4 : Pruebas de particionamiento de bases de datos NoSQL

Juan Sebastián Vacca Peña

ID 100098569

Corporación Universitaria Iberoamericana

Bases de Datos Avanzadas

13 diciembre de 2023

Contenido

Requerimientos funcionales	2
Escalabilidad	2
Rendimiento	2
Disponibilidad	2
Seguridad	2
Respaldo	2
Pruebas de particionamiento	3
Github	8

Requerimientos funcionales

Escalabilidad

La base de datos debe ser capaz de escalar horizontalmente, aumentando su tamaño en función al aumento de usuarios, sin generar interrupciones ni conflictos al momento de generar consultas.

Rendimiento

La base de datos deberá mantener un rendimiento eficiente manejando tráfico de diferentes magnitudes.

Disponibilidad

El sistema debe estar en funcionamiento continuo durante todo el evento, sin interrupciones, y estar disponible en todo momento para los usuarios.

Seguridad

La base de datos debe ser segura y confiable, para conservar temas de confidencialidad dentro del campeonato.

Respaldo

La base de datos tiene respaldo de información al ser replicada y alojada en nodos secundarios que garantizan la información de la base de datos.

Pruebas de particionamiento

1. Levantamiento del cluster: `cluster=new ShardingTest ({shards: 3, chunksize:1})`

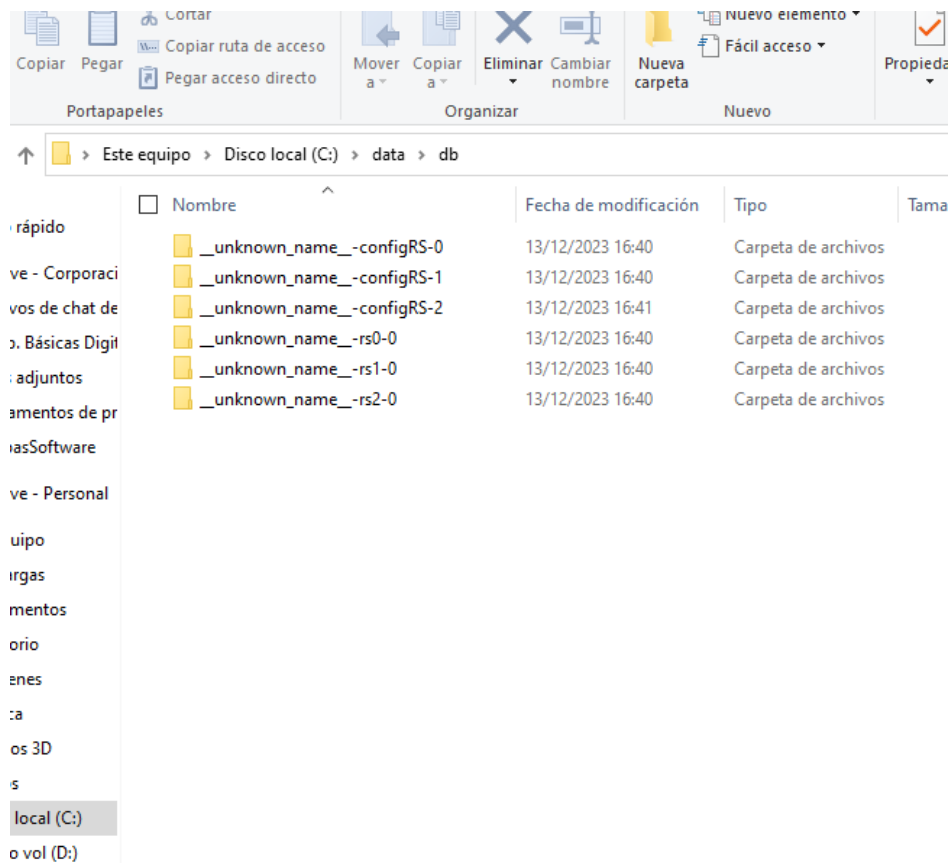
```

    return _callIsMaster() == false;
  }, msg, timeout);
},
  "getSecondaries" : function(timeout) {
    var master = this.getPrimary(timeout);
    var secs = [];
    for (var i = 0; i < this.nodes.length; i++) {
      if (this.nodes[i] != master) {
        secs.push(this.nodes[i]);
      }
    }
    return secs;
  },
  "getSecondary" : function(timeout) {
    return this.getSecondaries(timeout)[0];
  },
  "getArbiters" : function() {
    let arbiters = [];
    for (let i = 0; i < this.nodes.length; i++) {
      const node = this.nodes[i];

      let isArbiter = false;

      assert.retryNoExcept(() => {
        isArbiter = isNodeArbiter(node);
        return true;
      }, `Could not call 'isMaster' on ${node}.`, 3, 1000);
    }
  }
}

```



2. Hacemos ingreso de una cantidad de registros aleatoria en diferentes tablas alojadas en la base de datos "Futbolact_1". Con esto comprobamos el rendimiento de la base de datos con cantidades grandes de datos superiores a un dato, en este caso 50000 datos, 25000 y 20000 respectivamente.

```

mongo shell
MongoDB shell version v4.4.25
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("4526743c-882d-44ad-b113-787902949c0b") }
MongoDB server version: 4.4.25
---
The server generated these startup warnings when booting:
  2023-12-11T13:47:27.871-05:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
---
> db = (new Mongo("localhost:20006")).getDB("Futbolact_1")
uncaught exception: SyntaxError: missing ) after argument list :
@(<shell>):1:42
> db = (new Mongo("localhost:20006")).getDB("Futbolact_1")
Futbolact_1
mongos> for(i=0;i<50000;i++){
...   db.arbitros.insert({idarbitro:"idarbitro" +i, nombreArbitro:"Recuento de arbitros"
...   +i, date: new Date()});
... }

```

1 inserción de datos a tabla arbitro

```

mongos> for(i=0;i<25000;i++){
...   db.equipos.insert({name:"name" +i, numjugadores:"numjugadores"
...   +i, date: new Date()});
... }

```

2 inserción de datos a tabla equipos

```

WriteResult({ "nInserted" : 1 })
mongos> for(i=0;i<20000;i++){
...   db.goles.insert({equipo:"equipo" +i, goles:"Recuento de goles"
...   +i, date: new Date()});
... }

```

3 inserción de datos a tabla goles

Verificamos la inserción de los datos: `db.arbitros.count()`, `db.equipos.count()`, `db.goles.count()`, con esto verificamos el funcionamiento de la base de datos y la captura de las inserciones registradas anteriormente

```

mongos> db.arbitros.count()
50000
mongos> db.equipos.count()
25000
mongos>
mongos> db.goles.count()
20000
mongos>

```

3. Activación del sharding: para este proceso nos tenemos que redireccionar a la consola mongo Shell donde se creo el balanceador que alojamos en el puerto 20006 `shard1 = new Mongo("localhost:20006")`

```

mongo shell
mongos> shard1 = new Mongo("localhost:20006")
connection to localhost:20006
mongos>

```

4. Activamos el sharding en la base de datos: `sh.enableSharding("Futbolact_1")`

```

mongos> sh.enableSharding("Futbolact_1")
{
  "ok" : 1,
  "operationTime" : Timestamp(1702507751, 3),
  "$clusterTime" : {
    "clusterTime" : Timestamp(1702507751, 3),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
mongos>

```

5. Comprobamos la distribución de los datos

```

mongo shell
MongoDB shell version v4.4.25
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("79444805-ca9c-4fff-bc89-12e5e4157ebd") }
MongoDB server version: 4.4.25
---
The server generated these startup warnings when booting:
  2023-12-11T13:47:27.871-05:00: Access control is not enabled for the database. Read and write a
ccess to data and configuration is unrestricted
---
> shard1 =new Mongo("localhost:20000")
connection to localhost:20000
>

```

```

> shard1DB = shard1.getDB("Futbolact_1")
Futbolact_1
>

```

```

> shard1DB.arbitros.count()
50000
>

```

```

> shard2 = new Mongo("localhost:20001")
connection to localhost:20001
> shard2DB = shard2.getDB("Futbolact_1")
Futbolact_1
> shard2DB.arbitros.count()
0
>

```

6. Activación del shard

```
mongo shell
mongos> shard1 = new Mongo("localhost:20006")
connection to localhost:20006
mongos>
```

7. Probamos la existencias del shard y su estado

```
mongo shell
mongos> shard1 = new Mongo("localhost:20006")
connection to localhost:20006
mongos> sh.status()
--- Sharding Status ---
  sharding version: {
    "_id" : 1,
    "minCompatibleVersion" : 5,
    "currentVersion" : 6,
    "clusterId" : ObjectId("657a246a4b1d4575e8114b80")
  }
  shards:
    { "_id" : "__unknown_name__-rs0", "host" : "__unknown_name__-rs0/DESKTOP-IT871UI:20000", "state" : 1 }
    { "_id" : "__unknown_name__-rs1", "host" : "__unknown_name__-rs1/DESKTOP-IT871UI:20001", "state" : 1 }
    { "_id" : "__unknown_name__-rs2", "host" : "__unknown_name__-rs2/DESKTOP-IT871UI:20002", "state" : 1 }
  active mongoses:
    "4.4.25" : 1
  autosplit:
    Currently enabled: no
  balancer:
    Currently enabled: no
    Currently running: no
    Failed balancer rounds in last 5 attempts: 0
    Migration Results for the last 24 hours:
      No recent migrations
  databases:
    { "_id" : "Futbolact_1", "primary" : "__unknown_name__-rs0", "partitioned" : false, "version" : { "uuid" : UUID("80bedce9-97eb-4957-9c06-878634c0e3b1"), "lastMod" : 1 } }
    { "_id" : "config", "primary" : "config", "partitioned" : true }
```

8. Prueba del balanceador para determinar si está activo o no

```
mongo shell
mongos> sh.getBalancerState()
false
mongos>
```

9. Se coloca en marcha el balanceador y se vuelve a realizar su prueba de actividad

```

mongo shell
mongos> sh.getBalancerState()
false
mongos> sh.setBalancerState(true)
{
  "ok" : 1,
  "operationTime" : Timestamp(1702508599, 3),
  "$clusterTime" : {
    "clusterTime" : Timestamp(1702508599, 3),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
mongos>

mongos> sh.getBalancerState()
true
mongos>

```

10. Probamos el status del balanceador

```

active mongoses:
  "4.4.25" : 1
autosplit:
  Currently enabled: yes
balancer:
  Currently enabled: yes
  Currently running: no
  Failed balancer rounds in last 5 attempts: 0
  Migration Results for the last 24 hours:
    No recent migrations
databases:
  { "_id" : "Futbolact_1", "primary" : "__unknown_name__-rs0", "partitioned" : true, "version" : { "u
    id" : UUID("80bedce9-97eb-4957-9c06-878634c0e3b1"), "lastMod" : 1 } }
    Futbolact_1.arbitros
      shard key: { "idarbitro" : 1 }
      unique: false
      balancing: true
      chunks:
        __unknown_name__-rs0    1
        { "idarbitro" : { "$minKey" : 1 } } -->> { "idarbitro" : { "$maxKey" : 1 } } on : __unkn
        own_name__-rs0 Timestamp(1, 0)
        { "_id" : "config", "primary" : "config", "partitioned" : true }
mongos>

```


11. Prueba de shards activos

```

false
mongos> db.adminCommand({listshards: 1})
{
  "shards" : [
    {
      "_id" : "__unknown_name__-rs0",
      "host" : "__unknown_name__-rs0/DESKTOP-IT871UI:20000",
      "state" : 1
    },
    {
      "_id" : "__unknown_name__-rs1",
      "host" : "__unknown_name__-rs1/DESKTOP-IT871UI:20001",
      "state" : 1
    },
    {
      "_id" : "__unknown_name__-rs2",
      "host" : "__unknown_name__-rs2/DESKTOP-IT871UI:20002",
      "state" : 1
    }
  ],
  "ok" : 1,
  "operationTime" : Timestamp(1702509021, 3),
  "$clusterTime" : {
    "clusterTime" : Timestamp(1702509021, 3),
    "signature" : {
      "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAAA="),
      "keyId" : NumberLong(0)
    }
  }
}
mongos>

```

Github

<https://github.com/SebastianVacca/BD>