**ACTIVIDAD 3 - CONCEPTOS Y COMANDOS BÁSICOS DEL PARTICIONAMIENTO EN BASES DE DATOS NOSQL**

Jhon Mario Galarza López

Alejandro Gómez Patiño

William Ruiz Martínez



Ingeniería de Software, Facultad Ingeniería

Corporación Universitaria IBEROAMERICANA

Santa Fe de Bogotá

2023

# Documento descripción del escenario en que se necesitaría particionamiento y la especificación de los requerimientos no funcionales para torneo deportivo de futbol

Con el fin de dar cumplimiento al torneo deportivo de futbol se desarrolla el documento de requerimientos no funciones, estos requerimientos son de gran importancia para el desarrollo del sistema, de esta manera se garantizará que el sistema funcione de una forma correcta y así satisfacer las necesidades del usuario y que todas funciones como lo previsto.

Para cumplir con los requisitos de alta disponibilidad y redundancia, podemos utilizar la estrategia de particionamiento horizontal (sharding) de MongoDB. Para ello, podemos dividir los datos de manera equilibrada entre varios servidores (shards), lo que permitirá una mejor distribución del tráfico y un mayor rendimiento del sistema.

## Para implementar la partición horizontal, se deben seguir los siguientes pasos:

* **Configurar el conjunto de réplicas:** Se debe crear un conjunto de réplicas que contenga los servidores que actuarán como shards. Para ello, se puede utilizar el comando rs.initiate() para iniciar el conjunto de réplicas y rs.add() para agregar los miembros del conjunto.
* **Configurar el router (mongos):** Se debe crear un proceso de router (mongos) que actúe como un proxy para los clientes que se conecten a la base de datos. Para ello, se puede utilizar el comando mongos --configdb <configdb>, donde <configdb> es la dirección del conjunto de réplicas configurado anteriormente.
* **Configurar las bases de datos y colecciones:** Se debe crear la base de datos y las colecciones que se utilizarán en el sistema. Para ello, se puede utilizar el comando use <database> para seleccionar la base de datos y db.createCollection() para crear las colecciones.
* **Habilitar el particionamiento:** Se debe habilitar el particionamiento en la base de datos mediante el comando sh.enableSharding("<database>"). A continuación, se debe elegir una clave de particionamiento que se utilizará para dividir los datos entre los shards.
* **Definir los chunks:** Se deben definir los chunks, que son los fragmentos de datos que se asignarán a los shards. Para ello, se puede utilizar el comando sh.splitAt("<collection>", {"<key>": <value>}) para dividir la colección en chunks en función de la clave de particionamiento.
* **Asignar los chunks a los shards:** Se deben asignar los chunks a los shards disponibles mediante el comando sh.addShardTag("<shard>", "<tag>") para asignar una etiqueta a cada shard y el comando sh.addTagRange("<collection>", {"<key>": <min>}, {"<key>": <max>}, "<tag>") para asignar cada chunk a un shard.

Con estos comandos, se puede crear un entorno de particionamiento horizontal en MongoDB que cumpla con los requerimientos de redundancia y disponibilidad 24x7 para el torneo de futbol.

**Nota:** Es importante tener en cuenta que la implementación de la partición horizontal puede variar según las necesidades específicas del sistema y el modelo de datos utilizado.

**Ejemplo de comandos:**

1. Configurar el conjunto de réplicas:

rs.initiate()

rs.add("shard1.example.com")

rs.add("shard2.example.com")

1. Configurar el router (mongos):

mongos --configdb

1. Configurar las bases de datos y colecciones:

use mydb

db.createCollection("mycollection")

1. Habilitar el particionamiento:

sh.enableSharding("mydb")

sh.shardCollection("mydb.mycollection", {"\_id": "hashed"})

1. Definir los chunks:

sh.splitAt("mydb.mycollection", {"

Para crear los shards se utiliza el comando sh.addShard() seguido de la conexión al servidor y el puerto en el que se va a ejecutar el shard. Por ejemplo, si queremos agregar un shard con la dirección IP "192.168.0.1" y el puerto "27017" se utilizaría el siguiente comando:

sh.addShard("192.168.0.1:27017")

Este comando agrega el shard especificado al clúster de MongoDB.

Para definir el particionamiento de la colección se utiliza el comando sh.shardCollection() seguido del nombre de la base de datos, la colección y la clave de particionamiento. Por ejemplo, si queremos particionar la colección "partidos" de la base de datos "torneo" utilizando la clave "equipo\_local", se utilizaría el siguiente comando:

sh.shardCollection("torneo.partidos", {"equipo\_local": 1})

Este comando define que la colección "partidos" se particione utilizando la clave "equipo\_local" y distribuyendo los documentos de manera equilibrada entre los shards.

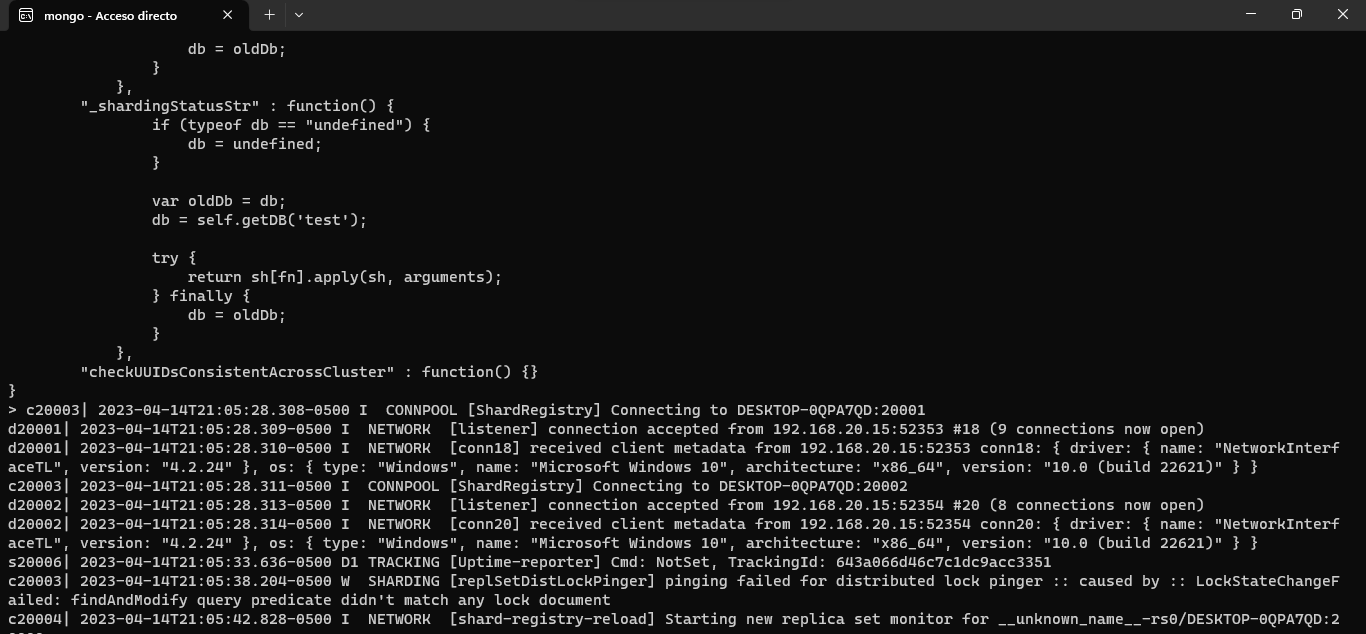
Además, se puede utilizar el comando sh.status() para ver el estado del clúster y verificar que los shards y la partición de la colección se hayan configurado correctamente.

Con estos comandos se puede crear el entorno de particionamiento definido para cumplir con los requerimientos planteados en el documento de requerimientos no funcionales.

**Creación de Particionamiento Horizontal**

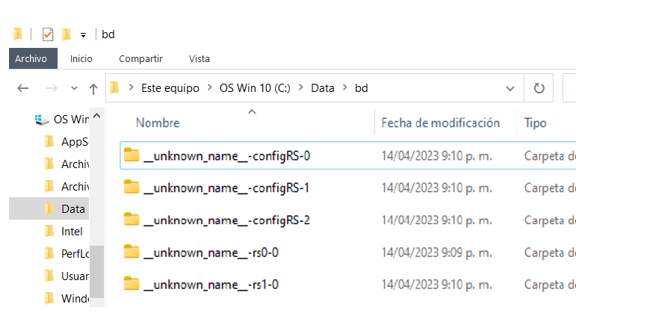
cluster=new ShardingTest ({shards: 3, chunksize:1})

Con la creación de este código se busca levantar el cluster con tres particiones, cada una con 1 megabyte de espacio.



Conexión correcta en el puerto 20001)

Se comprueba la creación de las instancias en la carpeta bd como se ve en la siguiente imagen:



Ahora abriremos otra ventada Shell, para hacer el proceso del balanceador para este proceso se usara el siguiente código:

bd = (new Mongo("localhost:20006")).getDB("Torneodeportivofutbol")

De esta manera se muestra el código que conectara a la base de datos donde se ingresa los registros.

Se deben ingresar los registros utilizando el siguiente código el cual está pidiendo que cree 1000 registros en la colección llamada torneo la cual creara e insertara en esa colección los datos que yo le pida en este caso será equipo y date que será la fecha del partido

for (i= 0; i < 1000; i++) { db.Torneo.insert({ equipo : "equipo " +i, date: new Date() }); }

por otro lado, podemos observar que en la otra consola la que está en listening nos muestra el proceso interno de la petición

Ahora se debe hacer la verificación de los registros ingresados:

db.Torneo.count()

db.Deportistas.count()

db.Equipos.count()

Texto, Correo electrónico

Descripción generada automáticamente

> shard3= new Mongo("localhost:20002")

> shard3DB = shard3.getDB("Torneodeportivofutbol")

> shard3DB.Torneo.count()

> shard3DB.Equipos.count()

> shard3DB.Deportistas.count()

Luego de verificar que no ha quedado la partición nos vamos a hacer la activación del Sharding en el balanceador.

Lo primero será hacer una comprobación del grupo de sharding utilizando status, entonces ejecutamos los siguientes comandos;

> shard1 = new Mongo("localhost:20006")

>sh.status()

Luego de ejecutarlo nos muestra la información del sharding (también muestra la Base de datos que queremos particionar), debemos fijarnos en el Currently eneable, ya que es el que nos dice si esta activado o no



De esta manera se evidencia que no se encuentra activado, entonces debemos proceder a activarlo,

Para activar el sharding utilizamos el siguiente comando;

sh.enableSharding("Torneodeportivofutbol")

Texto

Descripción generada automáticamente

Ahora debo decirle de que colección quiero que me haga la partición en este caso se hará con la colección Torneo, utilizamos el siguiente comando;

db.Torneo.ensureIndex({Torneo : 1})

De esta manera el código generara un index

Texto

Descripción generada automáticamente

Para hacer el proceso de particionamiento ejecutar el siguiente código;

sh.shardCollection("TorneoVoleibol.Torneo", {Torneo: 1})

Texto

Descripción generada automáticamente

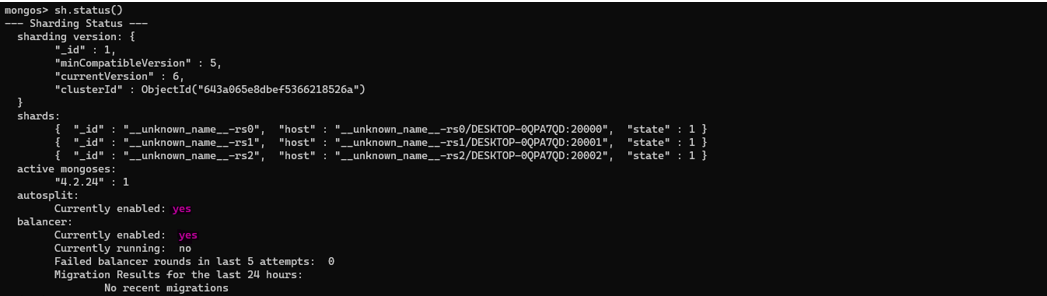
Verificamos cómo va el proceso en la parte de particiones debe estar en true y ya no en falso;



Para realizar la correcta partición ejecutamos el siguiente comando;

sh.setBalancerState(true)

de esta manera podemos ver que ya se encuentra activa la partición.



## Bibliografía

## Sarasa, A. (2016). Introducción a las bases de datos NoSQL usando MogoDB. Editorial UOC. <https://elibro.net/es/lc/biblioibero/titulos/58524>