Conceptos y Comandos básicos del particionamiento en bases de datos NoSQL - Requerimientos no funcionales

Natalia Stefania Gil Chaparro

Universidad Iberoamericana
Base de datos avanzadas
Tutor Jorge Castañeda
Diciembre 15, 2024

Contenido

1. Introducción

El presente documento especifica los criterios de calidad en cuanto a redundancia y disponibilidad 24x7 para la base de datos llamada "Copa Mundial". La base de datos se compone de los siguientes ficheros: Arbitros, Deportistas, EncuentrosDeportivos, Entrenadores, Equipos, Resultados y TablaPosiciones. Los criterios aquí descritos son esenciales para garantizar la fiabilidad y el rendimiento continuo del sistema durante el evento deportivo.

2. Desarrollo del tema

Disponibilidad

Disponibilidad 24x7:

- La base de datos debe estar disponible 24 horas al día, 7 días a la semana, sin interrupciones, para asegurar que los datos estén siempre accesibles para consultas y actualizaciones en tiempo real.
- Se implementará un sistema de alta disponibilidad que utilice la replicación de datos en múltiples nodos distribuidos geográficamente.

Redundancia

Redundancia de Datos:

- Los datos deben estar replicados en al menos tres nodos distintos para garantizar que no haya pérdida de datos en caso de fallo de un nodo.

Rendimiento

Rendimiento de Consultas y Escrituras:

- El tiempo de respuesta de las consultas no debe exceder los 100 milisegundos bajo carga normal y los 200 milisegundos bajo carga máxima.

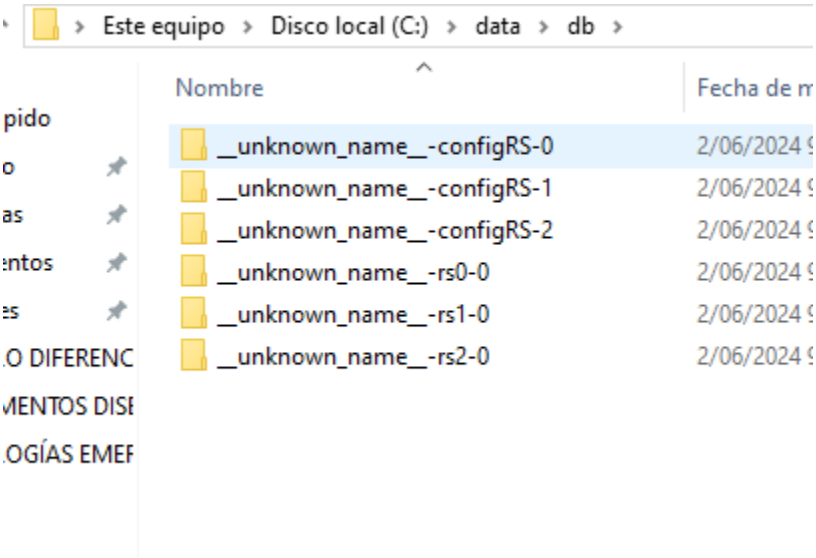
Desarrollo de comandos

1 Levantar el clúster

El primer paso que se ejecutó, fue el levantamiento del cluster al ejecutar el siguiente comando en la consola de mongoDB:

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

Luego de levantar el clúster se van a crear las siguientes instancias creadas en la ruta data/db



2 Inserción de datos

Abrimos una nueva consola para ejecutar el siguiente comando

```
db = (new Mongo("localhost:20006")).getDB("CopaMundial")
```

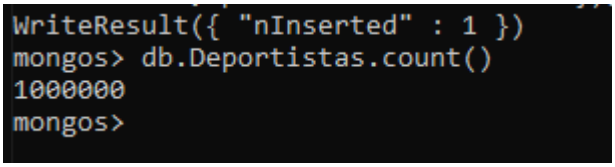
Con este comando entramos a la BD “CopaMundial”, para luego ejecutar el siguiente bucle:

```
CopaMundial
mongos> for (i= 0; i < 1000000; i++) {db.Deportistas.insert({nombre : "Daniel" +i, apellido: " Molina" +i,
edad: 25 +i, posicion: "Central" +i});}
```

Mediante este comando, estamos insertando 1000000 de registros al fichero “Deportistas”, luego de ejecutar este comando vamos observando que en la consola principal se inicia la inserción de los datos al fichero “Deportistas”

3 Revisión de los datos cargados

Cuando finaliza la inserción de los datos, podemos insertar el siguiente comando, para verificar que se encuentre registrados todos los datos que ingresamos.



4 Comprobación de la distribución de los datos

Debemos abrir una nueva consola, allí entraremos nuevamente a la BD CopaMundial y validaremos los registros que contiene cada uno de nuestros nodos, como se muestra a continuación:

```
---
> shard1 = new Mongo("localhost:20000")
connection to localhost:20000
> shard1DB = shard1.getDB("CopaMundial")
CopaMundial
> shard1DB.Deportistas.count()
0
> shard2= new Mongo("localhost:20001")
connection to localhost:20001
> shard2DB = shard2.getDB("CopaMundial")
CopaMundial
> shard2DB.Deportistas.count()
0
> shard3= new Mongo("localhost:20002")
connection to localhost:20002
>
> shard3DB = shard3.getDB("CopaMundial")
CopaMundial
> shard3DB.Autores.count()
0
> shard3DB.Deportistas.count()
1000000
>
```

Según esto podemos evidenciar que toda la información se encuentra registrada en el shard3

5 Ver estado sharding

Debemos volver a la consola donde ejecutamos el bucle, para que por medio del siguiente comando podamos validar el estado en el que se encuentra el sharding:

```
1000000
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("665d1a702bbdf57a94d95876")
  }
  shards:
        {  "_id" : "__unknown_name__-rs0",  "host" : "__unknown_name__-rs0/DESKTOP-
NTCHPTF:20000",  "state" : 1,  "topologyTime" : Timestamp(1717377660, 1) }
        {  "_id" : "__unknown_name__-rs1",  "host" : "__unknown_name__-rs1/DESKTOP-
NTCHPTF:20001",  "state" : 1,  "topologyTime" : Timestamp(1717377660, 6) }
        {  "_id" : "__unknown_name__-rs2",  "host" : "__unknown_name__-rs2/DESKTOP-
NTCHPTF:20002",  "state" : 1,  "topologyTime" : Timestamp(1717377660, 10) }
  active mongoses:
        "5.0.26" : 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" : "CopaMundial",  "primary" : "__unknown_name__-rs2",  "partitione
d" : false,  "version" : {  "uuid" : UUID("82ea37c5-5599-4a9f-9491-d227e151f3ff"),
 "timestamp" : Timestamp(1717378371, 1),  "lastMod" : 1 } }
        {  "_id" : "config",  "primary" : "config",  "partitioned" : true }
mongos>
```

6 Activar sharding

Como pudimos observar anteriormente el estado del balanceador no está activo, por lo tanto, debemos ejecutar el siguiente comando para activarlo.

7 Creación de índice

Debemos seleccionar un índice para la fragmentación, como lo realizamos a continuación:

```
mongos> sh.enableSharding("CopaMundial")
{
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1717386715, 1),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1717386715, 1)
}
mongos>
```

8 Activar compartimiento

Debemos activar el proceso para compartir la colección, como lo hacemos a continuación:

```
mongos> db.Deportistas.createIndex({nombre : 1})
{
        "raw" : {
                "__unknown_name__-rs2/DESKTOP-NTCHPTF:20002" : {
                        "numIndexesBefore" : 1,
                        "numIndexesAfter" : 2,
                        "createdCollectionAutomatically" : false,
                        "commitQuorum" : "votingMembers",
                        "ok" : 1
                }
        },
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1717387136, 3),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1717387136, 3)
}
mongos>
```

```
mongos> sh.shardCollection("CopaMundial.Deportistas", {nombre: 1})
{
        "collectionsharded" : "CopaMundial.Deportistas",
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1717387405, 23),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1717387405, 19)
}
mongos>
```

```
mongos> sh.status()
--- Sharding Status ---
  sharding version: {
        "_id" : 1,
        "minCompatibleVersion" : 5,
        "currentVersion" : 6,
        "clusterId" : ObjectId("665d1a702bbdf57a94d95876")
  }
  shards:
        {  "_id" : "__unknown_name__-rs0",  "host" : "__unknown_name__-rs0/DESKTOP-NTCHPTF:20000",  "state" : 1,  "topologyTime" : Timestamp(1717377660, 1) }
        {  "_id" : "__unknown_name__-rs1",  "host" : "__unknown_name__-rs1/DESKTOP-NTCHPTF:20001",  "state" : 1,  "topologyTime" : Timestamp(1717377660, 6) }
        {  "_id" : "__unknown_name__-rs2",  "host" : "__unknown_name__-rs2/DESKTOP-NTCHPTF:20002",  "state" : 1,  "topologyTime" : Timestamp(1717377660, 10) }
  active mongoses:
        "5.0.26" : 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" : "CopaMundial",  "primary" : "__unknown_name__-rs2",  "partitioned" : true,  "version" : {  "uuid" : UUID("82ea37c5-5599-4a9f-9491-d227e151f3ff"),
imestamp" : Timestamp(1717378371, 1),  "lastMod" : 1 } }
                CopaMundial.Deportistas
                        shard key: { "nombre" : 1 }
                        unique: false
                        balancing: true
                        chunks:
                                __unknown_name__-rs2    2
                        { "nombre" : { "$minKey" : 1 } } -->> { "nombre" : "Daniel549998" } on : __unknown_name__-rs2 Timestamp(1, 0)
                        { "nombre" : "Daniel549998" } -->> { "nombre" : { "$maxKey" : 1 } } on : __unknown_name__-rs2 Timestamp(1, 1)
        {  "_id" : "config",  "primary" : "config",  "partitioned" : true }
```

9 Activar balanceador

Para activar el balanceador primero debemos cambiar el estado de la siguiente forma:

```
mongos> sh.getBalancerState()
false
mongos> sh.setBalancerState(true)
{
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1717387519, 3),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        },
        "operationTime" : Timestamp(1717387519, 3)
}
mongos> sh.getBalancerState()
true
mongos>
```

Luego debemos iniciar el proceso de la siguiente forma:

```
mongos> sh.getBalancerState()
true
mongos> sh.isBalancerRunning()
false
mongos>
```

Podemos validar que el proceso se está ejecutando en la consola principal:


```
lo","attr":{"host":"DESKTOP-NTCHPTF:20004","replicaSet":"__unknown_name__-configRS","helloReply":"{ compression: [ \"snappy\", \"zstd\", \"zlib\" ], topologyVersion: {
processId: ObjectId('665d1a69791717dc6cce05ec'), counter: 6 }, hosts: [ \"DESKTOP-NTCHPTF:20003\", \"DESKTOP-NTCHPTF:20004\", \"DESKTOP-NTCHPTF:20005\" ], setName: \"__
unknown_name__-configRS\", setVersion: 5, ismaster: false, secondary: true, primary: \"DESKTOP-NTCHPTF:20003\", me: \"DESKTOP-NTCHPTF:20004\", lastWrite: { opTime: { ts
: Timestamp(1717387613, 1), t: 1 }, lastWriteDate: new Date(1717387613000), majorityOpTime: { ts: Timestamp(1717387613, 1), t: 1 }, majorityWriteDate: new Date(17173876
13000) }, configsvr: 2, maxBsonObjectSize: 16777216, maxMessageSizeBytes: 48000000, maxWriteBatchSize: 100000, localTime: new Date(1717387613898), logicalSessionTimeout
Minutes: 30, connectionId: 12, minWireVersion: 13, maxWireVersion: 13, readOnly: false, ok: 1.0, $gleStats: { lastOpTime: Timestamp(0, 0), electionId: ObjectId('0000000
0000000000000000') }, lastCommittedOpTime: Timestamp(1717387613, 1), $clusterTime: { clusterTime: Timestamp(1717387613, 1), signature: { hash: BinData(0, 0000000000000
0000000000000000000000000), keyId: 0 } }, $configTime: Timestamp(1717387613, 1), $topologyTime: Timestamp(1717377660, 10), operationTime: Timestamp(1717387613, 1) }"}
}
s20006| {"t":{"$date":"2024-06-02T23:06:53.959-05:00"},"s":"D1", "c":"-",        "id":4333221, "ctx":"ReplicaSetMonitor-TaskExecutor","msg":"RSM received successful hel
lo","attr":{"host":"DESKTOP-NTCHPTF:20005","replicaSet":"__unknown_name__-configRS","helloReply":"{ compression: [ \"snappy\", \"zstd\", \"zlib\" ], topologyVersion: {
processId: ObjectId('665d1a6c536ddf16ae5e0843'), counter: 5 }, hosts: [ \"DESKTOP-NTCHPTF:20003\", \"DESKTOP-NTCHPTF:20004\", \"DESKTOP-NTCHPTF:20005\" ], setName: \"__
unknown_name__-configRS\", setVersion: 5, ismaster: false, secondary: true, primary: \"DESKTOP-NTCHPTF:20003\", me: \"DESKTOP-NTCHPTF:20005\", lastWrite: { opTime: { ts
: Timestamp(1717387613, 1), t: 1 }, lastWriteDate: new Date(1717387613000), majorityOpTime: { ts: Timestamp(1717387613, 1), t: 1 }, majorityWriteDate: new Date(17173876
13000) }, configsvr: 2, maxBsonObjectSize: 16777216, maxMessageSizeBytes: 48000000, maxWriteBatchSize: 100000, localTime: new Date(1717387613959), logicalSessionTimeout
Minutes: 30, connectionId: 12, minWireVersion: 13, maxWireVersion: 13, readOnly: false, ok: 1.0, $gleStats: { lastOpTime: Timestamp(0, 0), electionId: ObjectId('0000000
0000000000000000') }, lastCommittedOpTime: Timestamp(1717387613, 1), $clusterTime: { clusterTime: Timestamp(1717387613, 1), signature: { hash: BinData(0, 0000000000000
0000000000000000000000000), keyId: 0 } }, $configTime: Timestamp(1717387613, 1), $topologyTime: Timestamp(1717377660, 10), operationTime: Timestamp(1717387613, 1) }"}
}
s20006| {"t":{"$date":"2024-06-02T23:06:57.195-05:00"},"s":"D1", "c":"NETWORK",  "id":4668132, "ctx":"ReplicaSetMonitor-TaskExecutor","msg":"ReplicaSetMonitor ping succ
ess","attr":{"host":"DESKTOP-NTCHPTF:20003","replicaSet":"__unknown_name__-configRS","durationMicros":1015}}
s20006| {"t":{"$date":"2024-06-02T23:06:57.197-05:00"},"s":"D1", "c":"NETWORK",  "id":4668132, "ctx":"ReplicaSetMonitor-TaskExecutor","msg":"ReplicaSetMonitor ping succ
ess","attr":{"host":"DESKTOP-NTCHPTF:20004","replicaSet":"__unknown_name__-configRS","durationMicros":1099}}
s20006| {"t":{"$date":"2024-06-02T23:06:57.305-05:00"},"s":"D1", "c":"NETWORK",  "id":4668132, "ctx":"ReplicaSetMonitor-TaskExecutor","msg":"ReplicaSetMonitor ping succ
ess","attr":{"host":"DESKTOP-NTCHPTF:20005","replicaSet":"__unknown_name__-configRS","durationMicros":881}}
s20006| {"t":{"$date":"2024-06-02T23:06:58.777-05:00"},"s":"D1", "c":"EXECUTOR", "id":23106,   "ctx":"ShardRegistry-330","msg":"Reaping this thread","attr":{"nextThread
RetirementDate":{"$date":"2024-06-03T04:07:28.777Z"}}}
s20006| {"t":{"$date":"2024-06-02T23:06:58.777-05:00"},"s":"D1", "c":"EXECUTOR", "id":23105,   "ctx":"ShardRegistry-330","msg":"Shutting down thread","attr":{"threadNam
e":"ShardRegistry-330","poolName":"ShardRegistry"}}
s20006| {"t":{"$date":"2024-06-02T23:06:58.780-05:00"},"s":"D1", "c":"SHARDING", "id":22726,   "ctx":"ShardRegistryUpdater","msg":"Reloading shardRegistry"}
s20006| {"t":{"$date":"2024-06-02T23:06:58.781-05:00"},"s":"D1", "c":"EXECUTOR", "id":23104,   "ctx":"ShardRegistry-331","msg":"Starting thread","attr":{"threadName":"S
hardRegistry-331","poolName":"ShardRegistry"}}
s20006| {"t":{"$date":"2024-06-02T23:06:58.781-05:00"},"s":"D1", "c":"TRACKING", "id":20164,   "ctx":"ShardRegistry-331","msg":"{trackingMetadata}","attr":{"trackingMet
adata":"Cmd: NotSet, TrackingId: 665d4162462e86d29db1e073"}}
s20006| {"t":{"$date":"2024-06-02T23:06:58.782-05:00"},"s":"D1", "c":"SHARDING", "id":22731,   "ctx":"ShardRegistry-331","msg":"Succesfully retrieved updated shard list
 from config server","attr":{"shardsNumber":3,"lastVisibleOpTime":{"ts":{"$timestamp":{"t":1717387616,"i":1}},"t":1}}}
s20006| {"t":{"$date":"2024-06-02T23:06:58.782-05:00"},"s":"D1", "c":"NETWORK",  "id":20157,   "ctx":"ShardRegistry-331","msg":"Started targeter","attr":{"connectionStr
ing":"__unknown_name__-rs0/DESKTOP-NTCHPTF:20000"}}
s20006| {"t":{"$date":"2024-06-02T23:06:58.782-05:00"},"s":"D1", "c":"NETWORK",  "id":20157,   "ctx":"ShardRegistry-331","msg":"Started targeter","attr":{"connectionStr
ing":"__unknown_name__-rs1/DESKTOP-NTCHPTF:20001"}}
s20006| {"t":{"$date":"2024-06-02T23:06:58.783-05:00"},"s":"D1", "c":"NETWORK",  "id":20157,   "ctx":"ShardRegistry-331","msg":"Started targeter","attr":{"connectionStr
ing":"__unknown_name__-rs2/DESKTOP-NTCHPTF:20002"}}
s20006| {"t":{"$date":"2024-06-02T23:06:59.897-05:00"},"s":"D1", "c":"TRACKING", "id":20164,   "ctx":"UserCacheInvalidator","msg":"{trackingMetadata}","attr":{"tracking
Metadata":"Cmd: NotSet, TrackingId: 665d4163462e86d29db1e075"}}
```

10 Validar el particionamiento de los datos

Debemos conectarnos a cada nodo, ingresar a la Bd copaMundial y contar los registros que hay dentro de cada nodo

```
mongos> shard1 = new Mongo("localhost:20000")
connection to localhost:20000
mongos> shard1DB = shard1.getDB("CopaMundial")
CopaMundial
mongos> shard1DB.Deportistas.count()
499999
mongos> shard2= new Mongo("localhost:20001")
connection to localhost:20001
mongos> shard2DB = shard2.getDB("CopaMundial")
CopaMundial
mongos> shard2DB.Deportistas.count()
0
mongos> shard3DB = shard3.getDB("CopaMundial")
ReferenceError: shard3 is not defined :
@(shell):1:1
mongos> shard3= new Mongo("localhost:20002")
connection to localhost:20002
mongos> shard3DB = shard3.getDB("CopaMundial")
CopaMundial
mongos> shard3DB.Deportistas.count()
500001
mongos>
```

Enlace vídeo de explicación:
Explicación particionamiento y pruebas.mp4

3. Conclusiones

Redundancia: Implementar la replicación de datos asegura que la base de datos pueda tolerar fallos de uno o varios nodos, manteniendo la integridad y disponibilidad de los datos.

Disponibilidad 24x7: El uso de sharding distribuye eficientemente la carga de trabajo entre múltiples nodos, mejorando el rendimiento y asegurando que la base de datos esté disponible en todo momento.

Con estos criterios y estrategias en lugar, la base de datos "Copa Mundial" estará equipada para manejar eficazmente grandes volúmenes de datos y asegurar una disponibilidad continua, cumpliendo con los requerimientos críticos del evento deportivo.