Documento de Scripts

Juan David Ríos Escudero, Erick Santiago Bustos guzmán

Universidad Iberoamericana

Bases de datos avanzadas

Profesor Jorge Castañeda

01/12/2024

1. **Introducción**

El presente documento tiene como objetivo proporcionar una descripción detallada de los pasos realizados para configurar el particionamiento horizontal (sharding) en MongoDB para la base de datos torneo, la cual contiene colecciones de eventos, deportistas y arbitros. Dado el volumen de datos que se espera manejar en el escenario de un torneo deportivo, se implementó una estrategia de particionamiento horizontal para garantizar que la base de datos pueda escalar, distribuir la carga de trabajo de manera eficiente y ofrecer un alto rendimiento en consultas a lo largo del evento.

En este documento se incluyen los comandos necesarios para llevar a cabo la configuración del sharding, incluyendo la creación de los shards, la configuración de los servidores de replicación, la habilitación de sharding en la base de datos y las colecciones, y la fragmentación de los datos. Además, se detallan los resultados obtenidos durante la implementación, incluyendo la verificación de la distribución de datos y la mejora en el rendimiento, que evidencian la efectividad del particionamiento horizontal en MongoDB.

1. **Scripts de particionamiento y resultados obtenidos**

Se ha decidido utilizar fragmentación por hash en el campo id de cada colección. La fragmentación por hash es una técnica que distribuye los documentos entre los shards de manera equilibrada, lo que ayuda a mejorar el rendimiento y la escalabilidad.

2. **Comandos ejecutados**

A continuación, se detallan los pasos realizados y los comandos ejecutados para configurar el sharding:

1. **Crear los directorios de datos para los shards:**
   Se crearon directorios para almacenar los datos de cada shard. Los comandos utilizados fueron:
   md c:\shard_data\shard1\data1
   md c:\shard_data\shard1\data2
   md c:\shard_data\shard1\data3
   md c:\shard_data\shard2\data1
   md c:\shard_data\shard2\data2
   md c:\shard_data\shard2\data3
   md c:\shard_data\shard3\data1
   md c:\shard_data\shard3\data2
   md c:\shard_data\shard3\data3

2. **Iniciar los servidores de los shards:**
   Se iniciaron los servidores de cada shard utilizando los comandos mongod correspondientes:
   start mongod.exe --shardsvr --port 26017 --dbpath "c:\shard_data\shard1\data1" --replSet shard1_replset
   start mongod.exe --shardsvr --port 26117 --dbpath "c:\shard_data\shard1\data2" --replSet shard1_replset

start mongod.exe --shardsvr --port 26217 --dbpath "c:\shard_data\shard1\data3" --replSet shard1_replset

3. **Configurar los conjuntos de réplicas para los shards:**
   Para cada shard, se configuró un conjunto de réplicas para asegurar la redundancia y alta disponibilidad. El comando utilizado fue:
   rs.initiate({
   _id: "shard1_replset",
   members: [
   { _id: 0, host: "localhost:26017" },
   { _id: 1, host: "localhost:26117" },
   { _id: 2, host: "localhost:26217" }
   ]
   });

4. **Configurar el servidor de configuración:**
   Se iniciaron tres servidores de configuración y se configuró su conjunto de réplicas con los comandos correspondientes.
   start mongod.exe --configsvr --port 47017 --dbpath "c:\shard_data\config_server1\data1" --replSet configserver1_replset

5. **Iniciar el router (mongos) y agregar los shards:**
   El router mongos se inició con el siguiente comando, y se añadieron los tres shards al clúster:
   start mongos.exe --configdb configserver1_replset/localhost:47017,localhost:47117,localhost:47217 --port 1000

   Después, se agregaron los shards con:
   sh.addShard("shard1_replset/localhost:26017,localhost:26117,localhost:26217")

6. **Habilitar sharding en la base de datos 'torneo':**
   Se habilitó el sharding en la base de datos torneo:
   sh.enableSharding("torneo")

7. **Fragmentar las colecciones utilizando hash:**
   Finalmente, las colecciones fueron fragmentadas por hash en el campo id:
   sh.shardCollection("torneo.evento", { id: "hashed" })
   sh.shardCollection("torneo.deportistas", { id: "hashed" })
   sh.shardCollection("torneo.arbitros", { id: "hashed" })

3. **Resultados obtenidos**

Se verificó la distribución de los datos en los shards utilizando el comando:

sh.status()

- El comando mostró que los documentos fueron distribuidos correctamente entre los shards de acuerdo con la fragmentación por hash.
- Se generaron 1000 documentos de prueba en cada colección para simular un volumen de datos realista. La distribución de los documentos en los shards fue equilibrada, mejorando el rendimiento de las consultas.

Conclusiones

La implementación del particionamiento horizontal (sharding) en MongoDB para la base de datos torneo permitió distribuir los datos de manera eficiente entre varios servidores. Esto mejora la escalabilidad, disponibilidad y rendimiento de la base de datos. La fragmentación por hash en el campo id fue efectiva para equilibrar la carga entre los shards.

Bibliografía

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

MongoDB Inc. (2024). Sharding in MongoDB. Retrieved from https://www.mongodb.com/docs/manual/sharding/