Actividad 4

Pruebas de particionamiento de bases de datos NoSQL

Yesid Alexander Jiménez Vivas

Corporación Universitaria Iberoamericana
Facultad de Ingeniería
Bases de datos avanzadas

Ingeniería de Software
Mtro. Jorge Castañeda

Bogotá, Colombia
15 de diciembre de 2024

INTRODUCCIÓN

La gestión de grandes volúmenes de datos generados en tiempo real, como los de un evento deportivo de gran escala, representa un reto significativo para los sistemas de bases de datos. Para garantizar la eficiencia, el rendimiento y la escalabilidad de dichos sistemas, se emplean técnicas de particionamiento horizontal, o sharding, que permiten distribuir los datos a través de múltiples servidores. Esta técnica no solo mejora la capacidad de manejo de datos, sino que también asegura una alta disponibilidad y reduce los tiempos de respuesta en consultas complejas.

Este documento tiene como objetivo especificar los casos de prueba necesarios para validar la implementación del particionamiento horizontal en una base de datos MongoDB, aplicada al caso de un evento deportivo. Los casos de prueba definidos en este documento están alineados con los requerimientos no funcionales previamente establecidos, que incluyen escalabilidad, desempeño, alta disponibilidad y consistencia eventual. A través de estos casos de prueba, se evaluará el comportamiento del sistema ante situaciones de alta carga de datos y tráfico, garantizando que el particionamiento horizontal funcione de acuerdo con las expectativas y requisitos del sistema.

Los casos de prueba aquí documentados incluyen pruebas de distribución de datos entre shards, desempeño de consultas, escalabilidad horizontal y la capacidad de mantener la disponibilidad del sistema ante fallos en los shards. El objetivo es asegurar que la solución de particionamiento implementada cumpla con los criterios de desempeño y escalabilidad necesarios para manejar la demanda de datos en tiempo real durante un evento deportivo.

Este enfoque de prueba servirá para proporcionar evidencia empírica de que el sistema está correctamente configurado para manejar grandes volúmenes de datos, responder eficientemente a las consultas y mantenerse disponible en todo momento.

CASOS DE PRUEBA PARA VALIDAR EL PARTICIONAMIENTO HORIZONTAL

Escenario para el particionamiento

En este caso práctico, estamos trabajando con una base de datos para gestionar un evento deportivo de gran escala. Dada la naturaleza de este evento, los datos generados en tiempo real son de gran volumen y deben gestionarse con una alta disponibilidad y capacidad de respuesta rápida. Para lograr esto, se ha implementado un sistema de particionamiento horizontal (sharding), que distribuye los datos entre múltiples servidores para garantizar que el sistema se mantenga escalable, disponible y eficiente incluso con grandes cantidades de usuarios y datos.

Requerimientos No Funcionales

1. **Escalabilidad:** El sistema debe ser capaz de manejar una gran cantidad de datos de manera eficiente a medida que el número de usuarios y el volumen de eventos crecen.
2. **Desempeño:** Las consultas deben ejecutarse en menos de 2 segundos, incluso con grandes volúmenes de datos.
3. **Alta Disponibilidad:** El sistema debe seguir funcionando incluso si uno de los servidores falla, gracias a la replicación de los datos en diferentes nodos.

4. **Consistencia Eventual:** Para asegurar la alta disponibilidad, algunas operaciones pueden operar con consistencia eventual.

Objetivo de los Casos de Prueba

Los casos de prueba deben validar que el particionamiento horizontal (sharding) está funcionando correctamente y cumple con los requerimientos de desempeño y escalabilidad. A continuación, se detallan los casos de prueba a ejecutar.

CASOS DE PRUEBA PARA VALIDAR EL PARTICIONAMIENTO

Caso de prueba 1: Validación de distribución de datos

- **Objetivo:** Validar que los datos se distribuyen correctamente entre los diferentes shards.
- **Descripción:** Después de insertar una cantidad significativa de datos en la colección de eventos deportivos, se debe verificar que los datos están distribuidos entre los shards, basándose en la clave de particionamiento (evento_id o timestamp).
- **Procedimiento:**
    1. Insertar datos en la colección eventos que cubran una variedad de evento_id.
    2. Ejecutar el comando sh.status() y observar cómo se distribuyen los datos entre los shards.
    3. Verificar que los datos están correctamente distribuidos entre los servidores.

Resultado esperado: Los datos deben estar distribuidos entre los shards de acuerdo con la clave de particionamiento.

Caso de prueba 2: Validación de desempeño de consultas

- **Objetivo:** Medir el tiempo de respuesta para consultas que involucren rangos de fechas o eventos.
- **Descripción:** Realizar una consulta que recupere datos dentro de un rango específico de timestamp o evento_id para asegurarse de que la consulta sea eficiente.
- **Procedimiento:**
    1. Insertar datos de eventos deportivos con timestamps de diferentes días.
    2. Ejecutar una consulta que recupere eventos dentro de un rango de tiempo determinado.
    3. Medir el tiempo de respuesta utilizando herramientas como mongotop o mongo en la consola.

Resultado esperado: La consulta debe ejecutarse en menos de 2 segundos, cumpliendo con los requisitos de desempeño.

Caso de prueba 3: Validación de escalabilidad

- **Objetivo:** Validar que el sistema puede escalar horizontalmente agregando más shards.
- **Descripción:** Agregar un nuevo shard al sistema y verificar que el sistema sigue funcionando correctamente al agregar datos y realizar consultas.
- **Procedimiento:**

1. Insertar una gran cantidad de datos en los shards existentes.
2. Agregar un nuevo shard al clúster de MongoDB.
3. Verificar que el sistema sigue funcionando correctamente y que los datos se distribuyen adecuadamente entre los shards existentes y el nuevo shard.

Resultado esperado: El sistema debe continuar funcionando sin interrupciones y los datos deben distribuirse correctamente entre los nuevos y los antiguos shards.

Caso de prueba 4: Validación de alta disponibilidad

- **Objetivo:** Verificar que el sistema sigue funcionando si un shard falla.
- **Descripción:** Simular una falla en un shard y verificar que el sistema sigue funcionando, manteniendo la disponibilidad de los datos a través de la replicación.
- **Procedimiento:**
  1. Configurar los shards con replicación.
  2. Apagar uno de los shards y realizar consultas sobre los datos.
  3. Verificar que las consultas siguen funcionando y que los datos se sirven desde el shard replicado.

Resultado esperado: El sistema debe seguir funcionando sin pérdida de datos ni interrupciones, gracias a la replicación.

EJECUTAR LOS CASOS DE PRUEBA Y GENERAR REPORTE DE RESULTADOS

Después de definir los casos de prueba, es necesario ejecutarlos en el entorno de MongoDB donde se ha implementado el particionamiento horizontal. Para cada caso, debes registrar los resultados obtenidos, incluyendo las métricas de desempeño (tiempos de respuesta), la distribución de los datos y cualquier anomalía o problema observado.

Informe de resultados:

- Caso 1: Descripción de la distribución de datos y el uso de sh.status().
- Caso 2: Resultados de las consultas y el tiempo de respuesta.
- Caso 3: Observaciones sobre la adición de un nuevo shard y la distribución de datos.
- Caso 4: Resultados de la validación de alta disponibilidad después de simular una falla en un shard.

CONCLUSIÓN

El uso de particionamiento horizontal (sharding) en sistemas de bases de datos es una estrategia clave para abordar los desafíos asociados con la gestión de grandes volúmenes de datos en tiempo real, como los generados en eventos deportivos de gran escala. A través de la implementación de sharding en MongoDB, se puede garantizar que el sistema sea escalable, eficiente y capaz de manejar picos de carga sin comprometer el rendimiento ni la disponibilidad de los datos.

Los casos de prueba documentados en este trabajo han permitido validar que el particionamiento horizontal funciona correctamente según los requerimientos no funcionales establecidos, tales como escalabilidad, desempeño, alta disponibilidad y consistencia eventual. Las pruebas realizadas, que incluyen la distribución de datos entre shards, la medición del tiempo de respuesta en consultas y la validación de la alta disponibilidad ante fallos, confirman que el sistema es capaz de manejar la carga esperada durante el evento, respondiendo a las consultas en tiempo real y manteniendo su operación sin interrupciones.

En resumen, los resultados obtenidos demuestran que la estrategia de particionamiento horizontal aplicada a la base de datos cumple con los objetivos de desempeño y escalabilidad, asegurando que el sistema sea robusto, flexible y capaz de soportar el creciente volumen de datos y usuarios. Este enfoque garantizará una experiencia de usuario fluida y confiable, incluso en condiciones de alta demanda, contribuyendo al éxito del evento deportivo y al rendimiento general del sistema.

REFERENCIAS

Sarasa, A. (2016). Introducción a las bases de datos NoSQL usando MongoDB. Editorial UOC. https://www.editorialuoc.com/introduccion-a-las-bases-de-datos-nosql-usando-mongodb.

Aramburu Cabo, M. J. y Sanz Blasco, I. (2012). Bases de datos avanzadas. D - Universitat Jaume I. Servei de Comunicació i Publicacions, http://hdl.handle.net/10234/48034.