Icono

Descripción generada automáticamente

**Actividad 3 - Conceptos y Comandos básicos del particionamiento en bases de datos NoSQL**

Luis Alfonso Cano Quintero

Ronald Buitrago Cardona

Faculta de ingeniería, Corporación Universitaria Iberoamericana

Bases de datos avanzadas

Jorge Castañeda

15 de diciembre de 2024

**Introducción**

En el contexto de grandes eventos deportivos, como los torneos internacionales de fútbol, ​​los repositorios deben procesar grandes cantidades de información, incluidos datos sobre equipos, jugadores, partidos, resultados, estadísticas y usuarios que acceden a estos datos en tiempo real. El sistema debe soportar miles de solicitudes simultáneas de organizadores, fanáticos y analistas. La partición horizontal (particionamiento) es necesaria para garantizar que el sistema cumpla con los requisitos de rendimiento, escalabilidad y disponibilidad mediante la distribución de datos entre múltiples nodos de bases de datos para distribuir la carga y mejorar la capacidad de respuesta.

**Escenario planteado**

1. Descripción del caso:

* Torneo de futbol con 6 equipos, partidos y personal arbitral.
* Cada partido incluye información detallada sobre jugadores, estadísticas y resultados.

El sistema debe soportar:

* Inserciones en tiempo real de datos del partido.
* Consultas simultáneas desde múltiples ubicaciones geográficas.

2. Problemas actuales:

* Alta latencia debido a la carga centralizada.
* Escalabilidad limitada a medida que crecen los datos.
* Riesgo de interrupciones por sobrecarga del servidor.

3. Justificación del sharding:

* Distribuir datos entre nodos mejora el rendimiento y asegura alta disponibilidad.
* Garantiza que cada nodo maneje solo una parte de la carga total, reduciendo cuellos de botella.

**Especificación de requerimientos no funcionales**

1. Desempeño

Tiempo de respuesta:

* Mantener un tiempo de respuesta inferior a 100 ms para consultas individuales en escenarios de carga alta.
* Garantizar un tiempo de respuesta promedio inferior a 300 ms para operaciones de escritura masiva durante los partidos.

Capacidad de transacciones:

* Soporte para picos de hasta 20,000 operaciones por segundo en momentos críticos, como la actualización en vivo de estadísticas al final de los partidos.
* Tiempos consistentes para operaciones de lectura y escritura en nodos distribuidos.

2. Escalabilidad

Escalabilidad horizontal:

* Posibilidad de agregar nuevos nodos sin interrupción del servicio.
* Asegurar que la arquitectura permita crecer hasta manejar más de 100 TB de datos en los próximos 5 años.

Balanceo automático:

* Redistribución automática de fragmentos entre nodos al agregar nuevos shards para evitar desbalanceo de carga.

Soporte para expansiones geográficas:

* Configuración adaptada para incluir centros de datos en nuevas regiones si se amplía la cobertura del torneo.

3. Disponibilidad

Alta disponibilidad:

* Tiempo operativo mínimo del 99.99%, lo que implica no más de 52.56 minutos de inactividad al año.

Replicación robusta:

* Cada shard debe tener al menos tres réplicas distribuidas en diferentes ubicaciones físicas para mitigar fallos regionales.

Manejo de fallos:

* Capacidad para recuperar automáticamente nodos en caso de fallos de hardware o software sin afectar el servicio.

4. Confiabilidad

Consistencia eventual:

* Asegurar que todas las réplicas se sincronicen dentro de un período máximo de 2 segundos después de una actualización.

Integridad de los datos:

* Uso de validaciones para evitar registros duplicados o inconsistencias entre shards durante inserciones en tiempo real.

Pruebas continuas:

* Implementar pruebas automáticas para validar la integridad de los datos después de balanceos y operaciones críticas.

5. Seguridad:

Control de acceso:

* Restricciones a nivel de usuario y rol para prevenir accesos no autorizados a datos sensibles, como información de jugadores.

Cifrado de datos:

* Cifrado de datos en tránsito y en reposo para proteger información durante transferencias entre nodos.

Auditoría:

* Registros detallados de accesos y cambios realizados en la base de datos para garantizar la trazabilidad.

6. Mantenibilidad:

Monitoreo continuo:

* Implementar herramientas como MongoDB Atlas o Prometheus para supervisar métricas clave (tiempo de respuesta, uso de CPU, latencia de red).

Actualizaciones sin interrupciones:

* Habilitar un sistema de actualizaciones para aplicar parches de software sin afectar la operación del sistema.

Soporte técnico:

* Establecer un equipo de soporte disponible 24/7 para resolver problemas críticos.

7. Compatibilidad:

Integración con otras plataformas:

* Asegurar que la base de datos pueda integrarse fácilmente con aplicaciones de análisis en tiempo real, como paneles estadísticos y servicios móviles.

Estandarización:

* Uso de estándares abiertos para facilitar la migración de datos si se requiere un cambio de tecnología en el futuro.

**Estrategia de particionamiento propuesta**

1. Id de partición:

Se utilizará el campo `id= configrs` para la creación de los servidores de configuración y “p1” “p2” “p3” para los conjuntos de replicación, ya que divide los datos de manera uniforme y evita accesos cruzados entre nodos.

2. Topología:

Mínimo de 3 shards distribuidos geográficamente para alta disponibilidad.

**Conclusiones**

1. El particionamiento horizontal mejoró significativamente el rendimiento de la base de datos, asegurando tiempos de respuesta consistentes.

2. La estrategia de fragmentación distribuyó eficientemente los datos, evitando cuellos de botella.

3. Se recomienda monitorear y ajustar la configuración a medida que crezca el volumen de datos o cambien los patrones de acceso.

**Referencias**

*Leer el capítulo VIII (Sharding) del libro Sarasa, A. (2016). Introducción a las bases de datos NoSQL usando MongoDB. Editorial UOC.*

*Hows, D., Plugge, E., Membrey, K., & Hawkins, P. (2015). MongoDB: The definitive guide: Powerful and scalable data storage (3rd ed.). O’Reilly Media.*