Lectura 3: Consideraciones en bases de datos NoSQL

Diego Granados Retana 2022158363

Bases de datos II

8 de setiembre, 2023

Responda las siguientes preguntas:

¿Es posible utilizar una base de datos SQL como una base de datos key-value?, ¿Cómo la implementaría? Comente las implicaciones de rendimiento

Yo creo que sí es posible utilizar es posible usar una base de datos SQL como un key-value. Tendría una tabla donde tengo una columna de Id y otra que es un free-text, que puede almacenar cualquier cosa (Golotiuk, 2022). Un ejemplo de una implementación en MySQL puede ser el siguiente.

CREATE TABLE IF NOT EXISTS `mydb`.`table1` (

`key` INT NOT NULL,

`value` BLOB NOT NULL,

PRIMARY KEY (`key`)

ENGINE = InnoDB;

También el value puede ser con un JSON en lugar de un blob. Un problema que puede tener esta implementación es que gastaría mucho espacio. Si uno necesita almacenar un número pequeño, igual estaría utilizando todo el espacio del blob. Si uno quisiera reducir el almacenamiento, se puede tener una tabla que sea solo para un tipo de datos, pero esto implicaría tener muchas tablas.

¿En qué consisten los datos polimórficos? Explique la razón por la cual estos son un buen caso de uso en bases de datos documentales.

Los datos polimórficos son los que pueden tener diferencias entre registros. Las bases de datos documentales son buenas para este tipo de datos porque tiene un esquema flexible y dinámico. En MongoDB se permite tener datos polimórficos en la misma colección (Alger y Coupal, 2022). Aunque los datos pueden tener campos diferentes, sí hay algunos comúnes, entonces se pueden juntos. Por ejemplo, se puede tener una colección de un catálogo de productos. Cada uno tiene características diferentes, pero todos tienen precio, nombre, marca, entre otros. Si quisiéramos realizar una base de datos relacional que almacene la misma información, tendríamos que utilizar el patrón de características variables, lo que implicaría tener una tabla de productos con la información en común, una tabla de características específicas y otra tabla que relacione las características con los productos. Esto haría que las consultas sean sumamente complejas. En una base de datos de key-value, sí podemos almacenar datos polimórficos, pero no la podríamos consultar fácilmente ya que solo se pueden hacer por su key. Una base de datos wide-column sí podría almacenar los datos polimórficos, pero es menos flexible que una base de datos documental, por lo que la documental sigue siendo mejor (MongoDB, s.f.).

Presente 5 ejemplos de sistemas/casos de uso que podrían soportar consistencia eventual, Explique

1. Sistemas distribuidos: Los sistemas distribuidos necesitan tener toda la información sincronizada y este proceso puede ser complejo y largo. Por esta razón, si se necesita que el sistema sea altamente disponible, para evitar que el sistema se detenga cada vez que se necesiten sincronizar los datos, se puede permitir la consistencia eventual, donde algunas réplicas del sistema no van a tener la información completa y consistente (Keboola, 2021).
2. Red social: No es necesario poder ver las últimas publicaciones de todas las personas, ya que muy posiblemente las que estén disponible cuando carguemos la página sean más que suficientes para el entretenimiento. Además, la información no es vital para nadie, no es que alguien va a perder dinero si no ve lo más nuevo (Hagar, 2018).
3. Internet Domain Name System: Los servidores del DNS no reflejan los más nuevos. Los valores se guardan en la caché y luego se replican en todos los directorios. Puede ser que la página de uno no aparezca inmediatamente, pero eventualmente sí va a aparecer en toda la Internet.
4. Agregar un producto al carrito: Plataformas como Amazon muestran la cantidad de productos que quedan disponibles. Puede ser que el número que le aparezca a uno diga que sí hay, pero cuando uno lo va a comprar, ya no. Aunque esto puede ser un inconveniente para el usuario, sería mucho más complicado que apareciera el número exacto disponible ya que se tendría que actualizar constantemente y sería más caro para la plataforma que la pérdida de esa compra, la cual además técnicamente no se perdió porque alguien sí compró el producto.
5. Almacenamiento en línea de archivos: Cuando uno crea un archivo que un servicio como OneDrive almacena en la nube, el archivo no va a ser inmediatamente cargado a esta. Esto es debido a que posiblemente el usuario va a modificarlo mucho, como escribir un documento en Word, por lo que si se sube con cada cambio se generaría un desperdicio de recursos al almacenar versiones no muy distintas entre sí. Por lo tanto, es posible que el archivo se suba en intervalos de tiempo constantes o cada vez que se haga un cambio sustancial. De esta forma, si en la máquina local se pierde el archivo, en la nube hay una versión respaldada, que puede no estar completamente actualizada, pero igualmente es mejor que perder el archivo completamente.

¿Por qué es importante que nativamente una base de datos NoSQL implemente un REST API?

Es muy importante porque el protocolo HTTP es un estándar muy popular y utilizado en una gran cantidad de aplicaciones. Aunque posiblemente es ideal tener drivers propios del lenguaje en el que uno esté trabajando, como Pymongo para Python, si no existe uno, se puede recurrir a los requests HTTP para realizar operaciones en la base de datos. Además, tienen otros beneficios (Bennet, 2023). Se pueden utilizar en aplicaciones de front-end, dan flexibilidad para trabajar con múltiples formatos, uno puede implementar un balanceador de carga para evitar sobrecargar la base de datos (como una cola), añadir nuestra propia lógica de negocios, permiten tener escalabilidad y se integran fácilmente con herramientas de inteligencia de negocios.

¿Por qué la geolocalización de las bases de datos NoSQL pueden ayudar a mantener leyes de Data sovereignty?

Data sovereignty se refiere a las leyes aplicables a los datos debido a que están ubicados físicamente en un país (Cloudian, 2022). La geolocalización ayuda a adherirse a las leyes ya que podemos tener datos en diferentes países, entonces tenemos que cumplir las leyes de ahí. Por ejemplo, en la Unión Europea está el General Data Protection Regulation (MongoDB, s.f.). Hay diferentes regulaciones y políticas dependiendo de que si la información está “at rest” o en tránsito. Existen leyes que no permiten que cierta información sea transferida de una ubicación. Uno puede utilizar la geolocalización también para evitar más bien

requerimientos legales, ya que como los países tienen diferentes leyes, las empresas pueden utilizar eso en ventaja suya. Por ejemplo, pueden evitar impuestos o simplemente facilitar el rendimiento de una consulta al guardar la información cerca de donde se va a necesitar. Por estas razones, es importante escoger bien la ubicación de dónde se van a almacenar los datos. Tenemos que investigar las restricciones de las posibles ubicaciones que el proveedor de la base de datos nos brinda.

Bibliografía

Alger, K. W., & Coupal, D. (2022, 23 septiembre). Building with Patterns: The Polymorphic Pattern | MongoDB. https://www.mongodb.com/developer/products/mongodb/polymorphic-pattern/ Bennett, T. (2023, 18 mayo). Direct Database Access vs. REST APIs: Pros and Cons for application connectivity - DreamFactory Software-blog. DreamFactory Software- Blog - API Management, Enterprise Integrations, Data Security and More. https://blog.dreamfactory.com/direct-database-access-vs-rest-apis-pros-and-cons-for-application-connectivity/#:~:text=REST%20(representational%20state%20transfer)%20APIs,s)%20users%20need%20to%20access. Cloudian. (2022, 19 agosto). Data Sovereignty in the cloud: Key considerations. https://cloudian.com/guides/data-protection/data-sovereignty-in-the-cloud-key-considerations/ Golotiuk, D. (2022, 5 agosto). Key-value storage on top of MySQL - DataDenys - Medium. Medium. https://medium.com/datadenys/key-value-storage-on-top-of-mysql-7b7e032caa44 Haldar, D. (2018, 5 agosto). System Design Interview Concepts – Eventual consistency. A CODER'S JOURNEY. https://www.acodersjourney.com/eventual-consistency/ Keboola. (2021, 20 agosto). What is eventual consistency and why should you care about it? Recuperado 9 de septiembre de 2023, de https://www.keboola.com/blog/eventual-consistency MongoDB. (s. f.). GDPR Compliance — MongoDB. https://www.mongodb.com/it-it/products/platform/trust/gdpr MongoDB. (s. f.-b). The Top 7 Considerations when Evaluating NoSQL Databases. https://www.mongodb.com/collateral/considerations-when-evaluating-nosql-databases