Quiz #2

Fecha: 30/09/2022
Autor: Deyan Sanabria Fallas #2021046131

Explique el concepto de Write Concern en MongoDB

El *write Concern* nos ayuda a definir el nivel de consistencia que quiero en MongoDB, el *Write Concern* se puede ver de la siguiente manera:

{w: <value>, j: <bool>, wtimeout: <value>}

w: Numero de nodos que debe afirmar que llego la operación de escritura y fue exitosa.

j: Si se esta satisfechoi con la consistencia y los datos van a escribir a disco.

wtimeout: Es el tiempo que debe haber pasado, desde la operación, para decir que tenemos consistencia.

Explique diferencias entre bases de datos NoSQL y SQL

- Entre las diferencias más claras que hay entre SQL y NoSQL es el como se almacenan los datos y es que las bases SQL suelen guardar datos de forma estructurada que recuerda mucho a una tabla, todos los registros de esas tablas tienen unas columnas fijas donde todos los registros/filas deben tener estos datos rellenados y no pueden quedar vacíos, en cambio, bases de datos SQL trabajan con archivos, en específico, archivos JSON, donde no existe una estructura de datos definida como tal, si no que esta semi-estructurada, cualquier dato que sea opcional o vacío puede ser omitido.
- Otra diferencia muy clara es el rendimiento y es que las bases de datos tienen las propiedades ACID, las cuales bajan mucho el rendimiento de las bases de datos SQL a comparación de las NoSQL, las cuales no cumplen con dichas reglas. ACID significa:
    - **A**tomicity: Ejecutar un conjunto de sentencias como si fueran una sola en un ciclo
    - **C**onsistency: Si ocurre un error en media sentencia, poder volver atrás. Primero se debe hacer atómica.
    - **I**solation: Transacciones Aisladas, no hay conflictos entre ellas
    - **D**urability: La transacción siempre debe perdurar. En caso de catástrofe debe ser capaz de recuperarse.
      Dichas propiedades .
- Algunos términos como Particiones en SQL se les llama Shards en NoSQL.
- El sistema de comunicación de la base de datos y un programa no es igual, en el caso de las SQL se usan queries SQL, para las NoSQL, algunas tienen su propio lenguaje el cual, algunas usan como intermediaron para traducir queries SQL para poder ejecutarlas en las mismas.

Desde un punto de vista de una base de datos de series de tiempo, ¿Porqué la localidad de datos es relevante para la escogencia del hardware a utilizar?

Las bases de datos de series de tiempo implementan un sistema en donde tienen diversos *"data tiers"*, como por ejemplo elasticsearch, el cual implementa data tiers como 'Hot', 'Warm', 'Frozen', entre otros. Estos datos van pasando a otro tipo de "almacen" con forme pasa el tiempo y entre menos son accedidos ya que a la hora de escoger hardware, podemos guardar datos poco accedidos y muy viejos en una maquina con pocos recursos que no necesiten mucha inversión económica.

La localidad de datos nos dice que existe probabilidad de que, si uso un dato, vaya a usar los datos de su alrededor y que, si accedimos a un dato, este puede que vuelva a ser accedido poco después.

Ambos conceptos se relacionan a la hora de escoger el hardware donde se montará las bases de datos porque hay que tener muy en claro como vamos a montar los data tiers y el hardware que requieren, todo esto dependerá de los datos que guardamos y su procesamiento. Si necesitamos una gran cantidad de datos para su procesamiento ocupamos un hardware capaz de tener esos datos almacenados en donde ocurrirá el procesamiento para su fácil acceso, por ende, hay que escoger muy bien cuales de esos data tiers nos sirven y en donde montarlos.

Explique el concepto de Federated Queries y el impacto que tienen estas en el rendimiento de bases de datos.

Las consultas federadas es una manera de enviar una consulta a todos los servidores de bases de datos, que almacenan los datos que necesito, para posteriormente unir el resultado de la ejecución de la consulta en cada servidor individual.

Esto tienen ciertas desventajas de rendimiento, la primera es que en algún lugar se tendrá que unir toda la información, por ende, el envió de los resultados, dependiendo de que tan grande sea va a tardar bastante y será tan lento como el servidor más lento del grupo.

La segunda desventaja sería el tiempo de procesado de la unión de los datos que también dependerá de donde se haga la unión de los datos el tiempo que se tarde, junto a la cantidad de datos que tenga que unir.

Al final, en general, se va a aumentar los tiempos de respuesta a los clientes y el uso de CPU del servidor principal encargado de la unión de los datos va a aumentar, lo que puede perjudicar a otros clientes.