BASE DE DATOS NO RELACIONAL.

1. ¿Qué es una Base de Datos No relacional?

Una base de datos no relacional, también conocida como NoSQL, es un tipo de base de datos que no utiliza el modelo de tablas y relaciones de las bases de datos relacionales. En su lugar, almacena datos en formatos como documentos, grafos, pares clave-valor o columnas.

Características principales:

Flexibilidad en el esquema: *No requieren un esquema fijo, lo que permite almacenar datos no estructurados o semiestructurados.*

Escalabilidad horizontal: *Pueden manejar grandes volúmenes de datos distribuyéndolos a través de múltiples servidores.*

Diversidad de modelos de datos: *Incluyen varios tipos como clave-valor, documentos, gráficos y en memoria.*

Alto rendimiento: *Están optimizadas para modelos de datos específicos y patrones de acceso, lo que mejora el rendimiento a gran escala.*

Ventajas:

Adaptabilidad: *Ideales para aplicaciones que manejan datos variados y en constante cambio.*

Desempeño: *Ofrecen un rendimiento superior en aplicaciones que requieren acceso rápido a grandes volúmenes de datos.*

Desventajas:

Consistencia: *No siempre garantizan las propiedades ACID (atomicidad, consistencia, aislamiento y durabilidad) que son comunes en las bases de datos relacionales.*

Las bases de datos no relacionales son especialmente útiles en aplicaciones modernas como redes sociales, análisis de grandes datos y sistemas de recomendación, donde la flexibilidad y la escalabilidad son cruciales

2. ¿Las BADA No SQL permiten operaciones CRUD?

Sí, las bases de datos NoSQL permiten operaciones CRUD (Crear, Leer, Actualizar, Eliminar). Aunque el enfoque y la implementación pueden variar según el tipo de base de datos NoSQL, todas ellas soportan estas operaciones fundamentales:

1. ***Crear (Create)**: Se pueden insertar nuevos datos en la base de datos.*
2. ***Leer (Read):** Se pueden consultar y recuperar datos almacenados.*
3. ***Actualizar (Update)**: Se pueden modificar datos existentes.*
4. ***Borrar (Delete)**: Se pueden eliminar datos de la base de datos.*

Ejemplos según el tipo de base de datos NoSQL:

- ***Clave-valor**: Utilizan una clave única para almacenar y recuperar valores. Ejemplo: **Redis.***
- ***Documentos**: Almacenan datos en documentos (generalmente JSON o BSON). Ejemplo: **MongoDB.***
- ***Columnas**: Organizan datos en columnas en lugar de filas. Ejemplo: **Cassandra.***
- ***Grafos**: Utilizan nodos y aristas para representar y consultar relaciones. Ejemplo: **Neo4j.***

Cada tipo de base de datos NoSQL tiene su propia API o lenguaje de consulta para realizar estas operaciones, adaptándose a las necesidades específicas de las aplicaciones que las utilizan.

3. ¿Qué es un Replica Set?

*Un Replica Set es un grupo de instancias de bases de datos que mantienen el mismo conjunto de datos, proporcionando redundancia y alta disponibilidad. En **MongoDB**, por ejemplo, un **Replica Set** permite que una base de datos continúe funcionando incluso si una de las instancias falla.*

Componentes de un Replica Set:

Nodo Primario: *Es el único nodo que recibe y registra todas las operaciones de escritura. Mantiene un registro de operaciones (oplog) que los nodos secundarios replican.*

Nodos Secundarios: *Replican el registro de operaciones del nodo primario y aplican estas operaciones a sus propios conjuntos de datos para mantener la consistencia. Si el nodo primario falla, uno de los nodos secundarios puede ser elegido como el nuevo primario.*

Árbitro (opcional): *Participa en las elecciones para elegir un nuevo nodo primario, pero no almacena datos. Es útil en situaciones donde se necesita un número impar de nodos para evitar empates en las elecciones.*

Ventajas de los Replica Sets:

Redundancia: *Al tener múltiples copias de los datos en diferentes servidores, se proporciona tolerancia a fallos.*

Alta disponibilidad: *Si el nodo primario falla, un nodo secundario puede ser promovido a primario, asegurando que el sistema siga funcionando.*

Escalabilidad de lectura: *Las operaciones de lectura pueden ser distribuidas entre los nodos secundarios, mejorando el rendimiento.*

Funcionamiento:

Escrituras: *Todas las operaciones de escritura son manejadas por el nodo primario.*

Replicación: *Los nodos secundarios replican las operaciones del nodo primario a través del registro de operaciones.*

Elecciones: *Si el nodo primario falla, los nodos secundarios realizan una elección para determinar el nuevo nodo primario.*

Los ***Replica Sets*** *son esenciales para garantizar la disponibilidad y la integridad de los datos en entornos de producción, especialmente en aplicaciones que requieren alta disponibilidad y tolerancia a fallos.*

4. ¿En BADA No SQL se escala verticalmente u horizontalmente?

Las bases de datos NoSQL se escalan principalmente de manera horizontal. Esto significa que, en lugar de mejorar el rendimiento añadiendo más recursos (como CPU o memoria) a un solo servidor (escalabilidad vertical), se añaden más servidores al sistema para distribuir la carga de trabajo.

Ventajas de la escalabilidad horizontal:

Capacidad de manejo de grandes volúmenes de datos: *Al distribuir los datos y las cargas de trabajo entre múltiples servidores, se puede manejar un mayor volumen de datos y tráfico.*

Alta disponibilidad: *Si un servidor falla, otros servidores pueden continuar operando, lo que mejora la disponibilidad del sistema.*

Flexibilidad: *Es más fácil y económico añadir nuevos servidores al sistema que mejorar significativamente un solo servidor.*

Ejemplos de bases de datos NoSQL que escalan horizontalmente:

MongoDB: *Utiliza sharding para distribuir datos entre múltiples servidores.*

Cassandra: *Diseñada para manejar grandes cantidades de datos distribuidos en muchos servidores.*

Redis: *Puede ser configurada en un clúster para distribuir la carga de trabajo.*

La escalabilidad horizontal es una de las razones por las que las bases de datos NoSQL son populares en aplicaciones que requieren manejar grandes cantidades de datos y tráfico, como redes sociales, análisis de big data y servicios en la nube.

5. ¿Cómo está compuesta una Base de Datos No SQL de documentos? ¿Usa tablas, filas, registros?

Una base de datos NoSQL de documentos, como MongoDB, está compuesta por colecciones y documentos. No utiliza tablas, filas ni registros. Los documentos son estructuras de datos similares a JSON que contienen pares clave-valor.

Componentes de una base de datos de documentos:

- ***Colecciones:*** *Equivalentes a las tablas en una base de datos relacional. Una colección es un grupo de documentos relacionados.*
- ***Documentos****: Equivalentes a las filas en una base de datos relacional. Cada documento es una unidad de datos independiente y se almacena en formato* ***JSON*** *(JavaScript Object Notation) o* ***BSON*** *(Binary JSON).*
- ***Campos****: Equivalentes a las columnas en una base de datos relacional. Cada documento contiene campos que almacenan datos. Los campos pueden contener diferentes tipos de datos, como cadenas, números, listas y objetos anidados.*

Características clave:

- ***Esquema flexible:*** *Los documentos en una misma colección no necesitan tener la misma estructura ni los mismos campos.*
- ***Anidamiento:*** *Los documentos pueden contener otros documentos o listas, permitiendo una representación más natural y jerárquica de los datos.*
- ***Consultas avanzadas:*** *A pesar de no usar SQL, las bases de datos de documentos ofrecen potentes capacidades de consulta y agregación.*

Esta flexibilidad y capacidad de manejar datos semiestructurados hacen que las bases de datos de documentos sean ideales para aplicaciones con requisitos de datos variados y en constante evolución.

6. ¿En qué casos conviene usar BADA Relacionales y BADA No SQL?

Bases de Datos Relacionales (SQL): *Son convenientes cuando se necesita integridad de datos, transacciones ACID y relaciones complejas entre datos. Ejemplos: sistemas financieros, ERP.*

Bases de Datos NoSQL: *Son adecuadas para aplicaciones que requieren alta escalabilidad, flexibilidad en el esquema y manejo de grandes volúmenes de datos no estructurados. Ejemplos: redes sociales, big data, aplicaciones en tiempo real.*

Casos para usar Bases de Datos Relacionales (RDBMS):

1. ***Consistencia y transacciones****: Cuando se necesita garantizar la consistencia de los datos y las transacciones ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad). Ejemplo:* ***sistemas bancarios.***
2. ***Estructura de datos fija****: Cuando los datos tienen una estructura bien definida y no cambian con frecuencia. Ejemplo:* ***sistemas de gestión de inventarios.***
3. ***Consultas complejas****: Cuando se necesita realizar consultas SQL complejas y uniones entre múltiples tablas. Ejemplo:* ***sistemas de informes empresariales.***
4. ***Integridad referencial****: Cuando es crucial mantener relaciones y restricciones entre diferentes conjuntos de datos. Ejemplo:* ***sistemas de gestión de relaciones con clientes (CRM).***

Casos para usar Bases de Datos NoSQL:

1. ***Escalabilidad horizontal****: Cuando se necesita manejar grandes volúmenes de datos y tráfico distribuyendo la carga entre múltiples servidores. Ejemplo:* ***redes sociales y aplicaciones web de alta demanda.***
2. ***Datos no estructurados o semiestructurados****: Cuando los datos no tienen una estructura fija o cambian con frecuencia. Ejemplo:* ***almacenamiento de documentos, datos de sensores.***
3. ***Alta disponibilidad y tolerancia a fallos****: Cuando se necesita que el sistema esté siempre disponible y pueda recuperarse rápidamente de fallos. Ejemplo:* ***aplicaciones de comercio electrónico.***

4. ***Rendimiento****: Cuando el rendimiento es crítico y se necesita acceso rápido a los datos. Ejemplo:* ***sistemas de recomendación en tiempo real.***
5. ***Modelos de datos específicos****: Cuando se trabaja con datos que se ajustan mejor a modelos no relacionales, como grafos o pares clave-valor. Ejemplo:* ***análisis de redes sociales, cachés en memoria.***