MODULO 04 - EJERCICIO 01-B

ALEXIS YURI M.

Crea un mapa visual (diagrama, infografía o esquema) que explique el Teorema de CAP y ubique diferentes tecnologías de bases de datos según sus prioridades (CP, CA, AP).

1. Definición del teorema de CAP.

El Teorema de CAP establece que, en un sistema de bases de datos distribuido, es imposible garantizar simultáneamente las siguientes tres propiedades: Consistencia, Disponibilidad y Tolerancia a Particiones.

2. Definición de cada propiedad del teorema de CAP.

- Consistencia (C): Cada lectura recibe la escritura más reciente o un error. En un sistema consistente, todos los nodos de la red tienen los mismos datos al mismo tiempo. Un ejemplo simple es una base de datos bancaria: si transfieres dinero de la cuenta A a la cuenta B, el saldo de la cuenta B debe reflejar la cantidad transferida inmediatamente para que una consulta a cualquier nodo del sistema muestre el mismo saldo actualizado.

- Disponibilidad (A): Cada solicitud recibe una respuesta (sin un error o tiempo de espera), pero sin la garantía de que contenga la escritura más reciente. En un sistema disponible, el sistema sigue funcionando y puede responder a las solicitudes, incluso si uno o más nodos están fuera de servicio. Un ejemplo es una red social: si se cae un servidor, los usuarios aún pueden ver las publicaciones, aunque algunas publicaciones muy recientes de otros servidores pueden tardar un poco en aparecer.

- Tolerancia a Particiones (P): El sistema sigue funcionando a pesar de las interrupciones de comunicación (fallos de red) entre los nodos. Este es el requisito más importante en un sistema distribuido, ya que las particiones de red son inevitables. Un ejemplo es una base de datos global de comercio electrónico: si la red entre el servidor de América y el de Europa falla, ambos servidores pueden seguir aceptando pedidos de sus respectivos continentes.

3. Ubicación de tecnologías en el triángulo de CAP.

El Teorema de CAP se representa como un triángulo con cada propiedad en un vértice, mostrando que cada sistema de base de datos distribuida solo puede elegir dos de las tres.

- Prioridad CA (Consistencia y Disponibilidad): Estos sistemas priorizan la consistencia y la disponibilidad, lo que los hace ideales para entornos donde las particiones de red son poco probables. Si hay una partición, el sistema dejará de funcionar para evitar inconsistencias. Un ejemplo claro es una base de datos relacional tradicional en un solo servidor, como *PostgreSQL*. No es un sistema distribuido, por lo que no puede tolerar particiones por diseño.

- Prioridad CP (Consistencia y Tolerancia a Particiones): Estos sistemas garantizan la consistencia y la tolerancia a las particiones. Cuando ocurre una partición, el sistema se vuelve no disponible para las peticiones que afectarían a los datos inconsistentes. Ejemplos de este tipo de bases de datos son *MongoDB y Redis* (en su configuración clúster o distribuida), que priorizan que la información sea siempre la correcta, aunque esto signifique no estar siempre disponible.

- Prioridad AP (Disponibilidad y Tolerancia a Particiones): Estos sistemas garantizan la disponibilidad y la tolerancia a las particiones, lo que los hace ideales para grandes sistemas distribuidos. Cuando ocurre una partición, los nodos siguen aceptando peticiones, lo que puede resultar en inconsistencia temporal de los datos. La consistencia se alcanza "eventualmente". Un ejemplo de esto es *Cassandra*, que está diseñada para una disponibilidad extrema, incluso si los datos no son consistentes de inmediato en toda la red.

4. Casos de uso que prioriza cada combinación.

- CP (Consistencia y Tolerancia a Particiones): Ideal para sistemas que no pueden permitirse la inconsistencia de datos. Casos de uso incluyen: sistemas de banca digital, procesamiento de transacciones, o cualquier aplicación donde la integridad de los datos es la máxima prioridad. Por ejemplo, en un banco, no puedes permitir que el saldo de una cuenta sea inconsistente.

- AP (Disponibilidad y Tolerancia a Particiones): Perfectos para sistemas a gran escala que necesitan estar siempre en línea y operar a través de múltiples centros de datos. Casos de uso incluyen: redes sociales (feeds de noticias), servicios de streaming (Netflix), y sistemas de comercio electrónico (donde un inventario ligeramente desactualizado es preferible a que el sitio web no cargue).

- CA (Consistencia y Disponibilidad): Se utilizan principalmente en arquitecturas monolíticas o en clústeres de alta fiabilidad dentro de una única red. Son la elección tradicional para la mayoría de las aplicaciones empresariales que no necesitan distribuirse globalmente y donde las transacciones ACID son cruciales.

5. Mapa Visual.

El mapa visual explica el Teorema de CAP, mostrando en cada vértice del triángulo las tres propiedades: Consistencia, Disponibilidad y Tolerancia a Particiones. Se colocan los nombres de las bases de datos (MongoDB, Cassandra, PostgreSQL, Redis, etc.) en cada lado común entre 2 propiedades.

