1. Introducción a Cassandra

- **¿Qué es Cassandra?** Apache Cassandra es una base de datos NoSQL distribuida y altamente escalable diseñada para manejar grandes volúmenes de datos a través de múltiples nodos, sin un punto único de falla.
- **Características principales:**
    - NoSQL: No utiliza un modelo relacional como SQL. Los datos se almacenan en un modelo de filas y columnas similar a tablas, pero con características más flexibles.
    - Distribuida: Los datos se distribuyen automáticamente entre los nodos de un clúster.
    - Escalabilidad horizontal: Se pueden añadir más nodos al clúster sin afectar el rendimiento.
    - Alta disponibilidad: Diseñada para no fallar, incluso si algunos nodos están caídos.
    - Rendimiento consistente: Cassandra está optimizada para lecturas y escrituras rápidas.
- **Historia:** Cassandra fue originalmente desarrollada en Facebook en 2008 para mejorar su sistema de mensajería. Más tarde, fue donada a Apache Software Foundation y se convirtió en un proyecto de código abierto.

---

2. Arquitectura de Cassandra

- **Clúster:** Un clúster es un grupo de nodos que trabajan juntos. Los nodos comparten los datos y se organizan en un anillo (ring), donde no hay un nodo central.
- **Nodos y replicación**: Cada nodo tiene una parte de los datos. Cassandra utiliza un modelo de replicación en el que los datos se copian en múltiples nodos para garantizar redundancia y tolerancia a fallos.
- **Particionado:** Los datos se distribuyen utilizando un algoritmo de hash. Cada fila tiene una clave primaria, y Cassandra utiliza esta clave para determinar en qué nodo se almacenan los datos.
- **Consistencia:** Cassandra permite elegir el nivel de consistencia mediante políticas configurables:
    - Consistencia eventual: Los datos se sincronizan con el tiempo entre los nodos (útil para escalabilidad y rendimiento).
    - Consistencia fuerte: Todos los nodos deben estar sincronizados antes de confirmar una operación (útil para precisión).

---

3. Modelo de datos

- **Keyspaces:** Es el equivalente a una base de datos en Cassandra. Define un espacio de nombres para las tablas y sus configuraciones, como la replicación.
- **Tablas:**
    - Las tablas en Cassandra tienen una estructura similar a las tablas relacionales, pero con importantes diferencias.
    - Cada tabla requiere al menos una clave primaria (que puede incluir varias columnas como clave compuesta).
    - Cassandra no soporta las relaciones (joins) entre tablas como en SQL.
- **Columnas y filas:** Las filas en Cassandra son esquema-flexible, es decir, no todas tienen que contener las mismas columnas.

---

4. Cómo funciona Cassandra internamente

- **Escritura:**
    - Cuando un cliente escribe datos, estos se almacenan primero en una memoria temporal llamada Commit Log.
    - Luego, los datos se almacenan en una estructura en memoria llamada Memtable.
    - Periódicamente, la Memtable se vacía en un archivo en disco llamado SSTable (Sorted String Table).
- **Lectura:**
    - Cassandra busca primero en la Memtable.
    - Si los datos no están allí, los busca en las SSTables usando índices.
- **Compacción:** Para optimizar el almacenamiento y la velocidad de lectura, Cassandra combina y organiza las SSTables en el disco.
- **Replicación:**
    - Cassandra replica los datos en varios nodos del clúster.
    - Puedes configurar el número de réplicas mediante la estrategia de replicación en el keyspace:
        - SimpleStrategy: Ideal para un clúster de un solo datacenter.
        - NetworkTopologyStrategy: Diseñada para clústeres distribuidos entre varios datacenters.

---

5. Ventajas y desventajas

Ventajas:

- Escalabilidad lineal.
- Alta disponibilidad.
- Soporte para Big Data.
- Rendimiento rápido y consistente.
- Ideal para aplicaciones en tiempo real (IoT, análisis en tiempo real, redes sociales).

Desventajas:

- No es adecuada para aplicaciones con relaciones complejas entre datos.
- Puede ser difícil de aprender al principio debido a su modelo no relacional.
- Configuración avanzada requiere experiencia.

---

6. ¿Cuándo usar Cassandra?

Cassandra es ideal para:

- Aplicaciones que manejan grandes volúmenes de datos.
- Sistemas distribuidos que requieren tolerancia a fallos y disponibilidad 24/7.
- Casos donde la escalabilidad horizontal sea clave.
- Aplicaciones que no requieren relaciones complejas entre datos.

Ejemplos:

- Sistemas de mensajería como WhatsApp.
- Registros de eventos en tiempo real.
- Plataformas de comercio electrónico para almacenar carritos de compras y pedidos.

¿Qué es Big Data?

Big Data son datos que son:

1. **Muy grandes (Volumen):** Imagina millones de personas viendo series en Netflix cada segundo.
2. **Muy variados (Variedad):** Datos de gustos de los usuarios, qué series ven, cuánto tiempo las ven, en qué dispositivo, etc.
3. **Muy rápidos (Velocidad):** Estos datos llegan todo el tiempo, y necesitamos procesarlos casi en tiempo real.

Netflix, por ejemplo, necesita analizar *muchos datos muy rápido* para recomendarte lo que te gusta.

¿Cómo encaja Cassandra en Big Data?

1. **Almacenar grandes volúmenes de datos:** Cassandra guarda todos los datos sobre lo que ves en Netflix, desde tu serie favorita hasta a qué hora pausaste un capítulo.
2. **Alta velocidad:** Cassandra es rápida, lo que significa que puede manejar miles de usuarios escribiendo y leyendo datos al mismo tiempo.
3. **Distribución:** Como Cassandra guarda los datos en muchos nodos, puede crecer y manejar más datos simplemente añadiendo más máquinas.

El truco de las recomendaciones: Algoritmos y Cassandra

Cuando Netflix te recomienda qué ver, ocurre algo increíble detrás de escena:

1. **Recolectar datos:**
    - Netflix usa Cassandra para guardar cosas como:
        - Qué series ves.
        - Cuánto tiempo las ves.
        - Si las pausas o las terminas.
        - Si te gusta o no (por ejemplo, al darle "like" o viendo si terminaste la serie).
2. **Procesar los datos (Big Data en acción):**
    - Los datos almacenados en Cassandra se envían a herramientas como Apache Spark o Hadoop.
    - Estas herramientas usan algoritmos complejos para analizar patrones en los datos:
        - ¿Qué tienen en común las personas que vieron la misma serie?

        - ¿Qué otras series suelen ver después?
3. **Crear un modelo de recomendaciones:**
    - Netflix usa un algoritmo llamado Filtrado Colaborativo. Este algoritmo compara tus gustos con los de otras personas:
        - "Si a Juan le gustó 'Stranger Things' y también le gustó 'The Witcher', probablemente a María (que vio 'Stranger Things') también le guste 'The Witcher'."
    - También usan algoritmos de aprendizaje automático (machine learning) para predecir qué te gustará basándose en tus hábitos.
4. **Mostrarte las recomendaciones:**
    - Una vez que el algoritmo decide qué mostrarte, la información se guarda nuevamente en Cassandra para que esté lista la próxima vez que entres.

Cómo Cassandra ayuda en este proceso:

1. **Rápido acceso a los datos:**
    - Cassandra puede manejar consultas rápidas para saber:
        - "¿Qué series vio este usuario?"
        - "¿Cuáles son las series más populares ahora mismo?"
2. **Escalabilidad:**
    - A medida que Netflix crece y más usuarios se conectan, solo tienen que añadir más nodos a Cassandra. Esto asegura que siempre funcione sin problemas.
3. **Disponibilidad:**
    - Aunque uno de los servidores de Netflix falle, Cassandra sigue funcionando porque los datos están replicados en otros nodos.
4. **Escritura eficiente:**
    - Cada vez que haces clic en "Reproducir", Cassandra guarda esa acción de manera eficiente, sin ralentizar el sistema.

Ejemplo práctico: Recomendaciones de Netflix

Imagina que hay tres usuarios: Juan, María y Pedro.

1. **Recolectar datos:**
   Cassandra guarda:
    - Juan vio "Stranger Things" y "The Witcher".
    - María vio "Stranger Things".
    - Pedro vio "The Witcher".
2. **Analizar patrones (Filtrado Colaborativo):**
    - El algoritmo nota que:

        - "Juan y María tienen gustos similares porque ambos vieron 'Stranger Things'."
        - "Juan también vio 'The Witcher', así que es probable que a María también le guste 'The Witcher'."
3. **Recomendar:**
    - Cuando María abre Netflix, le aparece: "¡Te puede gustar 'The Witcher'!".

El papel de los algoritmos complejos

Además del Filtrado Colaborativo, hay otros algoritmos importantes en Big Data:

1. **Clustering (Agrupamiento):**
    - Divide a los usuarios en grupos según sus gustos. Por ejemplo:
        - Grupo 1: Personas que aman el terror.
        - Grupo 2: Personas que prefieren comedia romántica.
2. **Regresión:**
    - Predice cuánto tiempo pasarás viendo una serie o qué tan probable es que termines una temporada.
3. **Redes Neuronales:**
    - Netflix usa redes neuronales para entender mejor patrones de datos y hacer predicciones más precisas.