Proyecto: Carga y Análisis Concurrente de Datos Masivos de Viajes en Taxi de NYC

Objetivos del Proyecto:

El objetivo principal de este proyecto es que los estudiantes demuestren su capacidad para integrar conocimientos de diseño de bases de datos relacionales y programación concurrente en Java para resolver un problema de ingeniería común: la ingesta y el análisis eficiente de grandes volúmenes de datos de transporte urbano.

Los objetivos específicos incluyen:

1. **Diseño de Bases de Datos:** Aplicar los principios de normalización (hasta la Tercera Forma Normal - 3FN) para diseñar un esquema de base de datos robusto y eficiente en MySQL, basado en los datos de viajes en taxi.
2. **Manejo de Datos Masivos:** Desarrollar una solución en Java para procesar un archivo CSV de grandes dimensiones (más de 3 millones de filas) de los registros de viajes en taxi de NYC.
3. **Optimización de E/S:** Implementar técnicas de programación concurrente para optimizar el tiempo de carga de los datos del CSV a la base de datos.
4. **Consultas Avanzadas:** Diseñar e implementar métodos concurrentes para realizar consultas complejas y analíticas sobre la base de datos.

Fases del Proyecto y Requisitos Técnicos

Paso 1: Adquisición y Preparación del Conjunto de Datos (CSV)

Requisito: El proyecto debe comenzar con el archivo CSV `yellow_tripdata_2025-02.csv` de los **NYC Yellow Taxi Trip Records** (registros de viajes en taxi amarillo de NYC, febrero de 2025), conteniendo **más de tres millones de filas**. Los datos están disponible en el sitio web de la Comisión de Taxis y Limusinas de NYC (TLC).

Estructura de Columnas Clave del CSV (Ejemplo):

Columna	Tipo de Dato	Descripción	Normalización
`VendorID`	`INT`	Identificador del proveedor de la TLC.	Redundante (Depende de la descripción del proveedor)
`tpep_pickup_datetime`	`DATETIME`	Fecha y hora de recogida.	Atributo de la entidad `Trip`
`tpep_dropoff_datetime`	`DATETIME`	Fecha y hora de destino.	Atributo de la entidad `Trip`
`passenger_count`	`INT`	Número de pasajeros.	Atributo de la entidad `Trip`
`trip_distance`	`FLOAT`	Distancia del viaje en millas.	Atributo de la entidad `Trip`
`RateCodeID`	`INT`	Código de tarifa (e.g., 1=Estándar, 2=JFK).	Redundante (Depende de la descripción de la tarifa)
`PULocationID`	`INT`	ID de la Zona de Recogida (Pick-up Location ID).	Redundante (Depende del nombre de la zona y el distrito)
`DOLocationID`	`INT`	ID de la Zona de Destino (Drop-off Location ID).	Redundante (Depende del nombre de la zona y el distrito)
`payment_type`	`INT`	Código del método de pago (e.g., 1=Tarjeta, 2=Efectivo).	Redundante (Depende de la descripción del pago)
`fare_amount`	`FLOAT`	Monto de la tarifa.	Atributo de la entidad `Trip`
`tip_amount`	`FLOAT`	Monto de la propina.	Atributo de la entidad `Trip`
`total_amount`	`FLOAT`	Monto total cobrado.	Atributo de la entidad `Trip`

Nota para el Estudiante: Se adjunta el archivo de **TLC Taxi Zone Lookup Table** (tabla de búsqueda de zonas de taxi) `taxi_zone_lookup.csv` para obtener los nombres de las zonas (`Zone`) y los distritos (`Borough`) a partir de los `LocationID` . También el diccionario de datos `data_dictionary_trip_records_yellow.pdf` que describe la estructura del CSV.

Paso 2: Diseño y Normalización de la Base de Datos (MySQL - 3FN)

Requisito: Diseñar y crear la base de datos en MySQL. El esquema debe estar normalizado a la **Tercera Forma Normal (3FN)** para eliminar dependencias transitivas y redundancia, asegurando la integridad de los datos.

Proceso de Normalización Sugerido:

1. **Identificación de Redundancias:** Las descripciones de `VendorID` , `RateCodeID` , `payment_type` , y la información de la zona (`Zone` , `Borough`) que depende de `PULocationID` / `DOLocationID` son dependencias funcionales que violan 2FN y 3FN.
2. **Creación de Tablas Normalizadas**.
3. **Implementación en MySQL:** Crear las sentencias `CREATE TABLE` con claves primarias, foráneas e índices adecuados.

Entregable: El script SQL (`schema.sql`) con la definición completa de la base de datos.

Paso 3: Carga Concurrente de Datos del CSV a la BD (Java)

Requisito: Implementar una aplicación en Java que utilice programación concurrente para leer el archivo CSV y cargar los datos en las tablas normalizadas de MySQL.

Conceptos Clave a Aplicar:

- **Patrón Productor-Consumidor:** Un hilo productor lee el CSV, múltiples hilos consumidores procesan las líneas, realizan la lógica de inserción y ejecutan **inserciones por lotes (Batch Inserts)** en JDBC.
- **Medición de Rendimiento:** Medir y reportar el tiempo total de carga para diferentes configuraciones de hilos.

Paso 4: Consultas Concurrentes y Simulación de Carga Distribuida (Java)

Requisito: Implementar métodos en Java que utilicen programación concurrente para ejecutar un conjunto de consultas analíticas complejas a la base de datos, simulando un entorno de alta demanda.

Conceptos Clave a Aplicar:

- **Ejecución Asíncrona:** Utilizar un `ExecutorService` y tareas para gestionar la ejecución concurrente de las consultas.
- **Pool de Conexiones:** Uso de un Pool de Conexiones JDBC para gestionar eficientemente las conexiones bajo alta concurrencia.

Consultas Analíticas Complejas (Ejemplos):

1. **Consulta 1: Análisis de Propina por Zona y Hora Pico:** Calcular el **monto promedio de propina** por **método de pago** y **Borough de recogida** para viajes realizados en un rango de horas específico (e.g., 7:00 AM - 9:00 AM).
2. **Consulta 2: Identificación de Rutas Rentables:** Identificar el **Top 5 de pares de zonas (Recogida-Destino)** que generan la **mayor tarifa promedio por milla** para viajes con más de 2 pasajeros.
3. **Consulta 3: Distribución de Viajes por Tipo de Tarifa:** Contar el número total de viajes por cada `RateCodeDescription` (e.g., JFK, Newark, Estándar) y calcular el porcentaje que representa del total de viajes.
4. **Consultas 4-8:** Deben ser generadas por los integrantes del equipo y debidamente explicadas en el informe de entrega.

Entregable: Código Java con el método principal que orquesta la carga de datos (Paso 3) y la ejecución de consultas concurrentes (Paso 4), incluyendo la medición de rendimiento.

Estructura del Informe de Entrega

El informe debe ser un documento técnico y reflexivo que demuestre la comprensión de los conceptos teóricos y las decisiones prácticas tomadas durante la implementación. Debe tener un mínimo de 10 páginas y un máximo de 15.

I. Introducción

1. **Objetivos del Proyecto:** Reafirmar los objetivos planteados.
2. **Contexto y Motivación:** Descripción del problema de la ingesta y análisis de datos de transporte urbano de NYC.
3. **Estructura del Informe:** Descripción concisa de las secciones siguientes.

II. Diseño de la Base de Datos

1. **Análisis del CSV y Dependencias Funcionales:**
   - Descripción de la estructura del CSV de viajes en taxi.
   - Identificación de las dependencias funcionales (parciales y transitivas) que justifican la normalización (e.g., `LocationID` -> `ZoneName` , `Borough`).
2. **Esquema Normalizado (3FN):**
   - Diagrama Entidad-Relación (DER) del esquema final (`Trips` , `Vendors` , `RateCodes` , `PaymentTypes` , `TaxiZones`).
   - Justificación detallada de cómo se alcanzó la 3FN para cada tabla.

III. Implementación de la Carga Concurrente (Paso 3)

1. **Arquitectura Concurrente:**
   - Diagrama de flujo o de componentes que ilustre el patrón **Productor-Consumidor**.
   - Justificación de la elección del número de hilos consumidores.
2. **Manejo de la Persistencia:**
   - Explicación del uso de **Batch Inserts** en JDBC.
   - Descripción de la lógica para manejar la inserción en las tablas de dimensiones (uso de caché concurrente).
3. **Análisis de Rendimiento:**
   - Tabla comparativa del tiempo de carga total para diferentes configuraciones de hilos (e.g., 1, 4, 8).
   - Gráfico de barras o líneas que visualice la ganancia de rendimiento (Speedup) obtenida por la concurrencia.

IV. Implementación de Consultas Concurrentes (Paso 4)

1. **Diseño de la Capa de Consultas:**
   - Explicación del uso del `ExecutorService` y el pool de conexiones.
2. **Detalle de las Consultas:**
   - Para cada consulta compleja implementada, incluir: la pregunta de negocio, la sentencia SQL utilizada y el tiempo de ejecución medido.

V. Conclusiones y Trabajo Futuro

1. **Conclusiones:** Resumen de los logros del proyecto y las lecciones aprendidas.
2. **Recomendaciones:** Identificación de las debilidades o áreas de mejora.
3. **Trabajo Futuro:** Propuestas para extender el proyecto.

VI. Declaración del uso de IA Generativa

1. **Declaramos haber hecho uso del sistema de IA Generativa (Nombre del sistema/herramienta IA y versión: ChatGPT, Gemini, Copilot…) para:**
2. **Ejemplos de instrucciones (prompts):**

Adjuntar

1. Código Fuente.
2. Script SQL de Creación de la BD.