Universidad Autónoma del Estado de Hidalgo

ICBI

REPORTE DE PRÁCTICA NO. 2.4

PRACTICA NODOS BDD FLOTILLAS

ALUMNO:

Morales Vázquez Rubén

1. Introducción

En esta practica abordaremos temas clave como la fragmentación vertical, una técnica de diseño de bases de datos que divide una tabla en segmentos más pequeños para mejorar la eficiencia; los procesos ETL (Extract, Transform, Load), esenciales para la integración y transformación de datos desde múltiples fuentes; el uso de SELECT + INTO FILE y LOAD, que facilitan la exportación e importación de datos entre sistemas; y finalmente, cómo realizar consultas con SELECT en tablas de dos bases de datos diferentes, permitiendo un análisis más completo y flexible. Estos conceptos son pilares para cualquier profesional que trabaje con grandes volúmenes de información.

2. Herramientas empleadas

Las herramientas que fueron necesarias para la realización de esta practica fueron las siguientes:

1. Repositorios de Github. Plataforma web donde se encuentran nuestros apuntes donde podemos reciclar información.
2. MySQL Workbench. Es la aplicación donde logramos establecer la base de datos en lenguaje SQL.
3. Online LaTeX Editor Overleaf. Es un sitio web en el cual podemos generar nuestros documentos.

3. Marco teórico

Fragmentación Vertical

La fragmentación vertical es una técnica de diseño de bases de datos que consiste en dividir una tabla en fragmentos (o subconjuntos) basados en columnas. En lugar de dividir las filas (como en la fragmentación horizontal), aquí se separan las columnas de una tabla en varias tablas más pequeñas. Cada fragmento contiene un subconjunto de las columnas de la tabla original, pero todas las filas.
La fragmentación vertical se usa principalmente para:

1. Mejorar el rendimiento: Al reducir el tamaño de las tablas, las consultas que acceden a solo algunas columnas son más rápidas.
2. Optimizar el almacenamiento: Si algunas columnas se usan con menos frecuencia, se pueden separar para ahorrar espacio.
3. Seguridad: Puedes separar columnas sensibles en una tabla diferente y controlar el acceso a ellas.

Sintaxis Fragmentación Vertical

```
1        CREATE VIEW nombre_vista AS
2        SELECT columna1, columna2, ...
3        FROM nombre_tabla;
4
```

Listing 1: Ejemplo de código MySQL

Procesos ETL (Extract, Transform, Load)

Los procesos ETL son un conjunto de operaciones que permiten extraer, transformar y cargar datos desde múltiples fuentes hacia un destino, como un almacén de datos o un sistema de análisis. Son fundamentales en la integración de datos, especialmente en entornos donde se manejan grandes volúmenes de información proveniente de sistemas heterogéneos.

1. **Extracción (Extract)** En esta fase, los datos se extraen desde diversas fuentes, como:
   - Bases de datos relacionales (MySQL, PostgreSQL, Oracle, etc.).
   - Archivos planos (CSV, Excel, JSON, XML).
   - APIs o servicios web.
   - Sistemas legacy o aplicaciones empresariales.

   El objetivo es recolectar datos crudos, sin modificar, para su posterior procesamiento.
2. **Transformación (Transform)** En esta etapa, los datos extraídos se transforman para cumplir con los requisitos del destino. Esto incluye: Limpieza de datos: Eliminar duplicados, corregir errores o manejar valores nulos.
   - Normalización: Ajustar los datos a un formato estándar (por ejemplo, fechas en un mismo formato).
   - Enriquecimiento: Agregar información adicional o calcular nuevos campos.
   - Agregación: Resumir datos para análisis (por ejemplo, sumar ventas por región).
   - Validación: Asegurar que los datos cumplan con reglas de negocio o integridad.
3. **Carga (Load)** Finalmente, los datos transformados se cargan en un sistema de destino, como:
   - Un data warehouse (por ejemplo, Amazon Redshift, Google BigQuery, Snowflake).
   - Una base de datos analítica.
   - Un lago de datos (data lake).
   - La carga puede ser:
   - Completa: Se reemplazan todos los datos existentes.
   - Incremental: Solo se añaden los datos nuevos o modificados.

Herramientas ETL Existen diversas herramientas para implementar procesos ETL, tanto comerciales como de código abierto:

- Comerciales: Informatica, Talend, Microsoft SSIS, IBM DataStage.
- Open Source: Apache NiFi, Pentaho, Airflow.
- Basadas en la nube: AWS Glue, Google Dataflow, Azure Data Factory.

Importancia de los procesos ETL

- Integración de datos: Permiten combinar información de diferentes fuentes en un solo lugar.
- Calidad de datos: Aseguran que los datos sean precisos, consistentes y confiables.
- Análisis eficiente: Facilitan la creación de informes y dashboards para la toma de decisiones.
- Automatización: Reducen el tiempo y esfuerzo manual en la gestión de datos.

SELECT + INTO OUTFILE

El comando SELECT + INTO OUTFILE en MySQL se utiliza para exportar el resultado de una consulta a un archivo externo en el servidor. Es útil cuando necesitas guardar datos en un formato legible, como un archivo CSV o texto plano, para su posterior análisis, compartir con otros sistemas o realizar copias de seguridad.

¿Para qué sirve?
Exportar datos a un archivo CSV o texto plano.

- Generar informes o backups de datos.
- Integrar datos con otras herramientas o sistemas que requieren archivos como entrada.
- Facilitar la migración de datos entre sistemas.

Sintaxis

```
SELECT columnas
INTO OUTFILE 'ruta_del_archivo'
[OPCIONES]
FROM tabla
[WHERE condiciones];

```

Listing 2: Ejemplo de código MySQL

LOAD

El comando LOAD en MySQL se utiliza para importar datos desde un archivo externo (como un CSV o un archivo de texto) hacia una tabla en la base de datos. Es especialmente útil cuando necesitas cargar grandes volúmenes de datos de manera eficiente, como en procesos de migración, integración de sistemas o actualización de bases de datos.

¿Para qué sirve?

- Importar datos desde archivos CSV o de texto.
- Cargar grandes volúmenes de datos de manera rápida y eficiente.
- Migrar datos desde otros sistemas o aplicaciones.
- Actualizar tablas con información externa.

Sintaxis

```
LOAD DATA [LOCAL] INFILE 'ruta_del_archivo'
INTO TABLE nombre_tabla
[FIELDS TERMINATED BY 'delimitador']
[ENCLOSED BY 'caracter']
[LINES TERMINATED BY 'delimitador']
[IGNORE numero_de_lineas]
[(columna1, columna2, ...)];

```

Listing 3: Ejemplo de código MySQL

Script de creación de nodos

```sql
CREATE DATABASE LCS1_Principal;
USE LCS1_Principal;

CREATE TABLE flotilla (
    flotillaId INT PRIMARY KEY,
    nombreEmpresa VARCHAR(100),
    gestorFlotilla VARCHAR(100),
    fechaCreacion DATE
);

CREATE TABLE vehiculo (
    vehiculoId INT PRIMARY KEY,
    flotillaId INT,
    tipo VARCHAR(50),
    modelo VARCHAR(50),
    marca VARCHAR(50),
    anio INT,
    estado VARCHAR(20),
    fechaVerificacion DATE
);

CREATE TABLE documento (
    documentoId INT PRIMARY KEY,
    vehiculoId INT,
    tipo VARCHAR(50),
    fechaVencimiento DATE,
    estado VARCHAR(20),
    rutaArchivo VARCHAR(255)
);

```

Listing 4: Creacion de nodo LCS1 Principal

```sql
CREATE DATABASE LCS2_Mantenimiento;
USE LCS2_Mantenimiento;

CREATE TABLE vehiculo (
    vehiculoId INT PRIMARY KEY,
    estado VARCHAR(20),
    fechaVerificacion DATE
);

CREATE TABLE mantenimiento (
    mantenimientoId INT PRIMARY KEY,
    vehiculoId INT,
    fechaServicio DATE,
    tipoServicio VARCHAR(100),
    descripcion VARCHAR(200),
    costo DECIMAL(10, 2),
    estado VARCHAR(20)
);
```

Listing 5: Creacion de nodo LCS2 Mantenimiento

```sql
CREATE DATABASE LCS3_Rutas;
USE LCS3_Rutas;

CREATE TABLE vehiculo (
    vehiculoId INT PRIMARY KEY,
    tipo VARCHAR(50),
    modelo VARCHAR(50),
    marca VARCHAR(50),
    anio INT
);

CREATE TABLE conductor (
    conductorId INT PRIMARY KEY,
    nombre VARCHAR(100),
    numeroLicencia VARCHAR(50),
    vencimientoLicencia DATE,
    estado VARCHAR(20)
);

CREATE TABLE ruta (
    rutaId INT PRIMARY KEY,
    vehiculoId INT,
    conductorId INT,
    horaInicio DATETIME,
    horaFin DATETIME,
    distancia DECIMAL(10, 2),
    ubicacionInicio VARCHAR(100),
    ubicacionFin VARCHAR(100),
    estado VARCHAR(20)
);

CREATE TABLE transaccionCombustible (
    transaccionId INT PRIMARY KEY,
    vehiculoId INT,
    conductorId INT,
    monto DECIMAL(10, 2),
    cantidad DECIMAL(10, 2),
    tipoCombustible VARCHAR(20),
    fechaTransaccion DATETIME,
    ubicacion VARCHAR(100)
);

```

Listing 6: Creacion de nodo LCS3 Rutas

Script de extracción de datos

```
use sistemagestionflotillas;
SELECT flotillaId, nombreEmpresa, gestorFlotilla, fechaCreacion
INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/flotilla.txt'
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
FROM flotilla;

SELECT vehiculoId, flotillaId, tipo, modelo, marca, anio, estado,
fechaVerificacion
INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/vehiculo.txt'
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
FROM vehiculo;

SELECT documentoId, vehiculoId, tipo, fechaVencimiento, estado, rutaArchivo
INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/documento.txt'
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
FROM documento;

```

Listing 7: Extraccion para nodo LCS1-Principal

```
SELECT vehiculoId, estado, fechaVerificacion
INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/vehiculomante.txt'
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
FROM vehiculo;
SELECT mantenimientoId, vehiculoId, fechaServicio, tipoServicio, descripcion,
costo, estado
INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/mantenimiento.txt'
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
FROM mantenimiento;

```

Listing 8: Extraccion para nodo LCS2-Mantenimiento

```
SELECT vehiculoId, tipo, modelo, marca, anio
INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/vehiculorutas.txt'
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
FROM vehiculo;

SELECT conductorId, nombre, numeroLicencia, vencimientoLicencia, estado
INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/conductor.txt'
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
FROM conductor;

SELECT rutaId, vehiculoId, conductorId, horaInicio, horaFin, distancia,
ubicacionInicio, ubicacionFin, estado
INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/ruta.txt'
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
FROM ruta;

SELECT transaccionId, vehiculoId, conductorId, monto, cantidad, tipoCombustible,
fechaTransaccion, ubicacion
INTO OUTFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/
transaccionCombustible.txt'
FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n'
FROM transaccionCombustible;

```

Listing 9: Extraccion para nodo LCS3-Rutas

Script de carga de datos

```
        use lcs1_principal;
        LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/flotilla.txt'
        INTO TABLE flotilla
        FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
        LINES TERMINATED BY '\n';

        LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/documento.txt'
        INTO TABLE documento
        FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
        LINES TERMINATED BY '\n';

        LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/vehiculo.txt'
        INTO TABLE vehiculo
        FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
        LINES TERMINATED BY '\n';

```

Listing 10: Carga de datos nodo LCS1-Principal

```
        use lcs2_mantenimiento;
        LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/mantenimiento.txt'
        INTO TABLE mantenimiento
        FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
        LINES TERMINATED BY '\n';

        LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/vehiculomante.txt'
        INTO TABLE vehiculo
        FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
        LINES TERMINATED BY '\n';

```

Listing 11: Carga de datos nodo LCS2-Mantenimiento

```
        use lcs3_rutas;
        LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/conductor.txt'
        INTO TABLE conductor
        FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
        LINES TERMINATED BY '\n';

        LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/ruta.txt'
        INTO TABLE ruta
        FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
        LINES TERMINATED BY '\n';

        LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/
    transaccionCombustible.txt'
        INTO TABLE transaccioncombustible
        FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
        LINES TERMINATED BY '\n';

        LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/vehiculorutas.txt'
        INTO TABLE vehiculo
        FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
        LINES TERMINATED BY '\n';

```

Listing 12: Carga de datos nodo LCS3-Rutas

Script de consulta de datos

```
SELECT c.conductorId, c.nombre, c.numeroLicencia, c.estado AS estado_conductor, v.tipo
AS tipo_vehiculo, v.modelo, v.marca
FROM LCS3_Rutas.conductor c
JOIN LCS1_Principal.vehiculo v
ON c.conductorId = v.vehiculoId
WHERE c.estado = 'Activo';
```

Listing 13: Ver Conductores con su Número de Licencia y Estado

```
SELECT d.documentoId, d.tipo AS tipo_documento, d.fechaVencimiento, d.estado AS
estado_documento, v.vehiculoId, v.tipo AS tipo_vehiculo, v.modelo, v.marca
FROM LCS1_Principal.documento d
JOIN LCS1_Principal.vehiculo v
ON d.vehiculoId = v.vehiculoId
WHERE d.estado = 'Vigente';
```

Listing 14: Ver Documentos Vigentes con Información del Vehículo

Scripts de Triggers

```
        use lcs1_principal;

        DELIMITER //
        CREATE TRIGGER after_insert_vehiculo
        AFTER INSERT ON LCS1_Principal.vehiculo
        FOR EACH ROW
        BEGIN
            -- Insertar en LCS2_Mantenimiento.vehiculo
            INSERT INTO LCS2_Mantenimiento.vehiculo (vehiculoId, estado, fechaVerificacion)
            VALUES (NEW.vehiculoId, NEW.estado, NEW.fechaVerificacion);

            -- Insertar en LCS3_Rutas.vehiculo
            INSERT INTO LCS3_Rutas.vehiculo (vehiculoId, tipo, modelo, marca, anio)
            VALUES (NEW.vehiculoId, NEW.tipo, NEW.modelo, NEW.marca, NEW.anio);
        END //
        DELIMITER ;

```

Listing 15: Insertar un registro

```
DELIMITER //
CREATE TRIGGER after_update_vehiculo
AFTER UPDATE ON LCS1_Principal.vehiculo
FOR EACH ROW
BEGIN
    -- Actualizar en LCS2_Mantenimiento.vehiculo
    UPDATE LCS2_Mantenimiento.vehiculo
    SET estado = NEW.estado,
        fechaVerificacion = NEW.fechaVerificacion
    WHERE vehiculoId = NEW.vehiculoId;

    -- Actualizar en LCS3_Rutas.vehiculo
    UPDATE LCS3_Rutas.vehiculo
    SET tipo = NEW.tipo,
        modelo = NEW.modelo,
        marca = NEW.marca,
        anio = NEW.anio
    WHERE vehiculoId = NEW.vehiculoId;
END //
DELIMITER ;

```

Listing 16: Actualizar un registro

```
DELIMITER //
CREATE TRIGGER after_delete_vehiculo
AFTER DELETE ON LCS1_Principal.vehiculo
FOR EACH ROW
BEGIN
    -- Eliminar en LCS2_Mantenimiento.vehiculo
    DELETE FROM LCS2_Mantenimiento.vehiculo
    WHERE vehiculoId = OLD.vehiculoId;

    -- Eliminar en LCS3_Rutas.vehiculo
    DELETE FROM LCS3_Rutas.vehiculo
    WHERE vehiculoId = OLD.vehiculoId;
END //
DELIMITER ;

```

Listing 17: Eliminar un registro

4. Conclusiones

En esta práctica, lo que desarrollamos fue lograr establecer una base de datos distribuidas pequeña con tres nodos en los cuales nos basamos en la base de datos de flotillas, utilizamos el proceso ETL, en el cual tenemos que utilizar y experimentar con la consulta de SELECT INTO OUTLFILE y LOAD el cual consiste en extraer datos de una base de datos y lograr insertarlos en una nueva base de datos, logrando asi la base de datos distribuida con datos insertados de otras bases de datos.

References

[1] Giménez, J. A. (**2019**). *Buenas prácticas en el diseño de bases de datos.* ARANDU UTIC, *6*(1), 193–210.

[2] Mosquera Palacios, D. J., & Wanumen Silva, L. F. (**2018**). *Arquitectura para la generación de consultas SQL usando lógica de conjuntos.* Visión Electrónica, *2*, 307–318.

[3] Date, C. J. (**2004**). An Introduction to Database Systems *8th ed Addison-Wesley*

[4] Elmasri, R. Navathe, S. B. (**2016**). Fundamentals of Database Systems *11th ed Pearson*