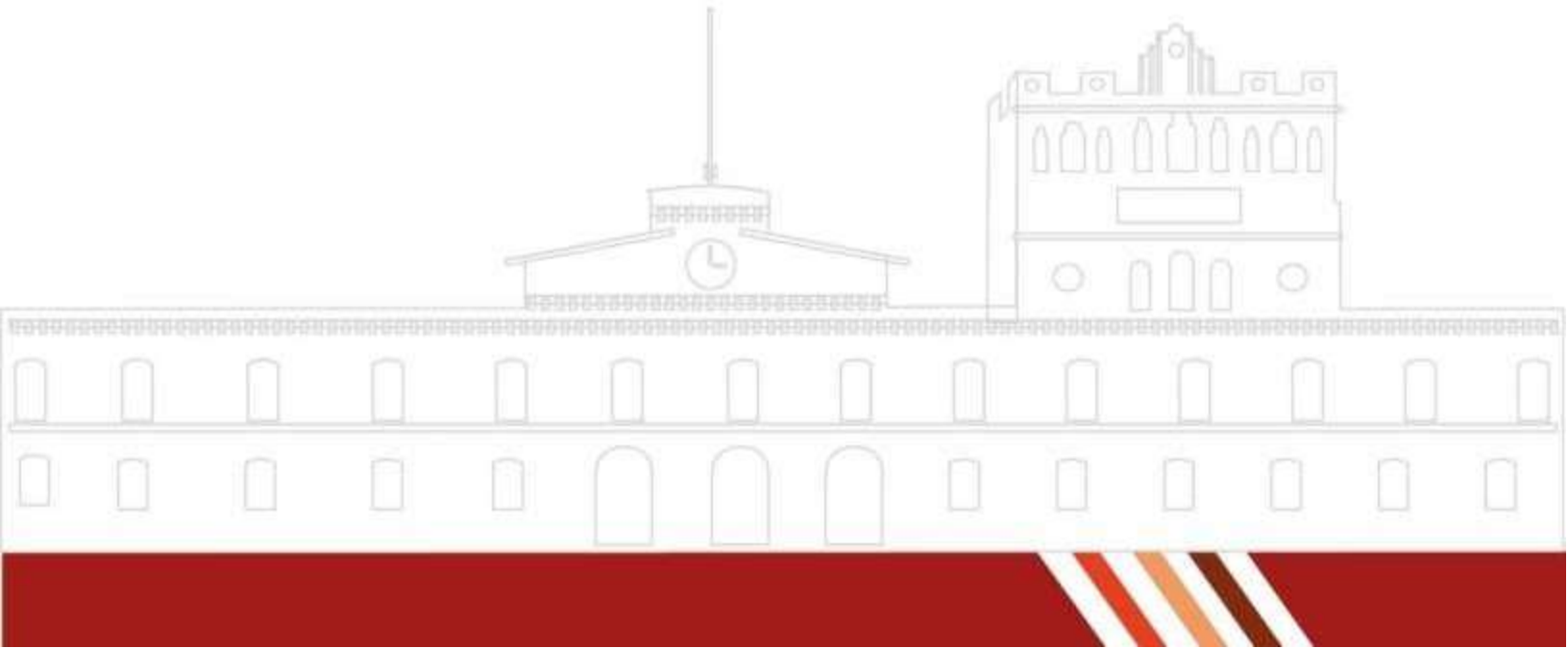
Reporte de Practica No. 2.1

Nombre de la Practica: Fragmentos

Alumno: Angel Amaya Zumaya

Dr. Eduardo Cornejo Velázquez



1. Introducción

La administración eficiente de una flotilla de vehículos es un pilar fundamental para el éxito operativo y financiero de empresas en sectores como logística, transporte y distribución. Una gestión inadecuada puede derivar en costos operativos elevados, incumplimiento de normativas y una reducción significativa en la productividad. La clave para superar estos desafíos reside en la centralización y el análisis de la información a través de un sistema de base de datos robusto y bien estructurado.

El presente reporte detalla el proceso de diseño e implementación de una base de datos relacional en MySQL, concebida para administrar de manera integral una flotilla de autos. El objetivo principal es crear una solución centralizada que no solo almacene datos, sino que también garantice su integridad, seguridad y disponibilidad para los distintos roles involucrados en la operación: administradores, talleres mecánicos, operadores y propietarios.

Para lograrlo, se ha seguido una metodología formal de diseño de bases de datos, partiendo del levantamiento de requisitos basados en un análisis de la industria, seguido por la creación de un modelo conceptual (Entidad-Relación), un modelo lógico normalizado y, finalmente, la implementación física en MySQL. Adicionalmente, se ha implementado un esquema de seguridad basado en fragmentación virtual a través de vistas SQL, asegurando que cada usuario acceda únicamente a la información pertinente a su rol.

2. Marco teórico

2.1. Bases de Datos Relacionales

Una base de datos relacional es un sistema de almacenamiento que organiza los datos en tablas compuestas por filas (tuplas) y columnas (atributos). Este modelo, propuesto por Edgar F. Codd, se fundamenta en la teoría de conjuntos y la lógica de predicados. Sus principales ventajas son la integridad de los datos, la flexibilidad para realizar consultas complejas mediante el lenguaje SQL (Structured Query Language) y la capacidad de minimizar la redundancia a través de la normalización.

2.2. MySQL

MySQL es uno de los Sistemas de Gestión de Bases de Datos Relacionales (RDBMS) de código abierto más populares del mundo. Es conocido por su fiabilidad, alto rendimiento y facilidad de uso. Al ser compatible con el estándar SQL, permite una implementación robusta de modelos relacionales, garantizando la atomicidad, consistencia, aislamiento y durabilidad (ACID) de las transacciones.

2.3. Metodología de Diseño de Bases de Datos

El diseño de una base de datos es un proceso estructurado que se divide en tres fases principales:

1. **Diseño Conceptual:** Se identifican las principales entidades, sus atributos y las relaciones entre ellas. El resultado es un Modelo Entidad-Relación (ERD), que es independiente del RDBMS a utilizar.
2. **Diseño Lógico:** El modelo conceptual se traduce en un esquema de base de datos, definiendo tablas, claves primarias (PK) y claves foráneas (FK). En esta fase se aplica el proceso de **normalización**.
3. **Diseño Físico:** Se toman decisiones de implementación específicas del RDBMS, como la definición de índices, tipos de datos y configuración de almacenamiento para optimizar el rendimiento.

2.4. Normalización

La normalización es el proceso de organizar las columnas y tablas de una base de datos para minimizar la redundancia de datos. Su objetivo es dividir las tablas grandes en tablas más pequeñas y bien estructuradas. Las formas normales (1NF, 2NF, 3NF, etc.) son un conjunto de reglas que guían este proceso. Para este proyecto, el diseño se adhiere a la Tercera Forma Normal (3NF), asegurando que todos los atributos de una tabla dependan únicamente de la clave primaria.

2.5. Fragmentación y Vistas (Views)

La fragmentación de bases de datos consiste en dividir una base de datos en piezas más pequeñas, o fragmentos. Puede ser horizontal (dividiendo filas) o vertical (dividiendo columnas). Si bien la fragmentación física se usa en sistemas distribuidos, una implementación virtual se puede lograr mediante **Vistas SQL**. Una vista es una consulta almacenada que se presenta como una tabla virtual. Son una herramienta de seguridad eficaz, ya que permiten restringir el acceso de los usuarios a un subconjunto específico de datos sin otorgarles permisos sobre las tablas base.

3. Metodología de diseño e implementación

3.1. Levantamiento de Requisitos

Basado en el análisis del artículo "Flotilla de autos: ¿cómo administrarla de forma eficiente?" de Edenred México, se identificaron los siguientes requisitos funcionales y de datos:

- **Entidades Clave:** Vehículo, Conductor, Propietario, Taller Mecánico, Mantenimiento, Documento, Gasto y Ruta.
- **Procesos a Gestionar:**
  - Control de gastos de combustible y peajes.
  - Gestión y seguimiento de vencimientos de documentos (seguros, tenencias, etc.).
  - Planificación y registro de mantenimientos preventivos y correctivos.
  - Asignación y seguimiento de rutas a los conductores.
- **Roles y Requisitos de Acceso:**
  - **Administrador de Flotilla:** Requiere una visión global de la operación, incluyendo el estado de todos los vehículos, rutas activas, costos totales y alertas de mantenimiento o vencimientos.

- **Taller Mecánico:** Solo necesita acceso a los datos técnicos de los vehículos que atiende y su historial de mantenimiento, sin ver información financiera o de rutas.
- **Operador (Chofer):** Necesita consultar las rutas que tiene asignadas, los vehículos que opera y registrar los gastos asociados.
- **Propietario del Auto:** Requiere visibilidad sobre el estado de sus vehículos, los costos de mantenimiento y los documentos asociados.

3.2. Modelo Conceptual

Se diseñó un Modelo Entidad-Relación que representa las entidades y sus interconexiones.

[Imagen de un Diagrama Entidad-Relación para un sistema de flotillas]

- Un Propietario puede poseer uno o muchos Vehiculos.
- Un Vehiculo es asignado a una o muchas Rutas a lo largo del tiempo.
- Un Conductor es asignado a una o muchas Rutas.
- Un Vehiculo recibe uno o muchos Mantenimientos en un TallerMecanico.
- Un Vehiculo tiene asociados muchos Documentos y Gastos.

3.3. Modelo Lógico y Normalización

El modelo conceptual se tradujo en el siguiente esquema relacional, aplicando la 3NF para evitar redundancias. Por ejemplo, los datos del conductor no se repiten en cada ruta, sino que se enlazan mediante una clave foránea (idConductor).

- **Propietario** (idPropietario, nombre, rfc, direccion, telefono)
- **Vehiculo** (idVehiculo, placa, marca, modelo, anio, numeroSerie, idPropietario)
- **Conductor** (idConductor, nombre, numeroLicencia, fechaVencimientoLicencia)
- **TallerMecanico** (idTaller, nombre, direccion, telefono)
- **Mantenimiento** (idMantenimiento, idVehiculo, idTaller, fechaServicio, descripcion, costo)
- **Documento** (idDocumento, idVehiculo, tipoDocumento, fechaVencimiento)
- **Gasto** (idGasto, idVehiculo, idConductor, concepto, monto, fecha)
- **Ruta** (idRuta, idVehiculo, idConductor, origen, destino, fechaSalida, estatus)

3.4. Seguridad y Operación

La estrategia de seguridad se basa en el **principio de mínimo privilegio**. El acceso a los datos no se otorga directamente sobre las tablas base. En su lugar, se crearon vistas SQL específicas para cada rol. Los permisos de la base de datos (GRANT SELECT) se asignarán únicamente sobre estas vistas, garantizando que los usuarios solo puedan consultar la información estrictamente necesaria para sus funciones.

4. Desarrollo (MySQL)

4.1. Esquema

El siguiente código SQL DDL (Data Definition Language) crea la estructura completa de la base de datos `gestion_flotilla`.

```
CREATE DATABASE IF NOT EXISTS gestion_flotilla;
USE gestion_flotilla;

CREATE TABLE Propietario (
  idPropietario INT PRIMARY KEY AUTO_INCREMENT,
  nombre VARCHAR(100) NOT NULL,
  rfc VARCHAR(13) UNIQUE
);

CREATE TABLE Vehiculo (
  idVehiculo INT PRIMARY KEY AUTO_INCREMENT,
  placa VARCHAR(10) UNIQUE NOT NULL,
  marca VARCHAR(50),
  modelo VARCHAR(50),
  anio INT,
  numeroSerie VARCHAR(20) UNIQUE,
  idPropietario INT,
  FOREIGN KEY (idPropietario) REFERENCES Propietario(idPropietario)
);
```

4.2. Datos de Ejemplo y Consultas (Fragmentos)

Se poblaron las tablas con datos de ejemplo utilizando sentencias `INSERT INTO`. Posteriormente, se crearon las vistas para cada rol.

Fragmento del Administrador

Esta vista consolida la información operativa clave.

```
CREATE OR REPLACE VIEW Vista_Administrador AS
SELECT
    v.placa, v.marca, v.modelo,
    c.nombre AS nombreConductor,
    r.origen, r.destino, r.estatus AS estatusRuta,
    m.fechaServicio, m.tipoMantenimiento
FROM Vehiculo v
LEFT JOIN Ruta r ON v.idVehiculo = r.idVehiculo
LEFT JOIN Conductor c ON r.idConductor = c.idConductor
LEFT JOIN Mantenimiento m ON v.idVehiculo = m.idVehiculo;

-- Consulta de ejemplo:
SELECT * FROM Vista_Administrador WHERE estatusRuta = 'En Curso';
```

Fragmento del Taller Mecánico

Esta vista muestra solo la información técnica y de mantenimiento de los vehículos atendidos por un taller específico (ej. Taller con ID 1).

```
CREATE OR REPLACE VIEW Vista_TallerMecanico_1 AS
SELECT
    v.placa, v.marca, v.modelo, v.anio, v.numeroSerie,
    m.fechaServicio, m.descripcion, m.costo
FROM Mantenimiento m
JOIN Vehiculo v ON m.idVehiculo = v.idVehiculo
WHERE m.idTaller = 1;

-- Consulta de ejemplo:
SELECT * FROM Vista_TallerMecanico_1 ORDER BY fechaServicio DESC;
```

5. Conclusiones

Se ha diseñado e implementado con éxito una base de datos relacional en MySQL que satisface los requisitos para una gestión integral de flotillas de vehículos. El modelo de datos normalizado garantiza la integridad y minimiza la redundancia, mientras que la arquitectura de seguridad basada en vistas SQL proporciona un acceso controlado y segmentado para cada tipo de usuario.

La solución propuesta constituye una base sólida y escalable. Centraliza la información crítica, facilitando la toma de decisiones estratégicas para el administrador, agilizando las operaciones para conductores y talleres, y ofreciendo transparencia a los propietarios.

Como trabajo futuro, se recomienda el desarrollo de una aplicación web o móvil que funcione como interfaz gráfica para interactuar con la base de datos. Esto simplificaría la entrada de datos y permitiría la creación de dashboards y reportes visuales para un análisis más intuitivo de la información.

Bibliografía

- Codd, E. F. (1970). A Relational Model of Data for Large Shared Data Banks. *Communications of the ACM,* 13(6), 377-387.

- Edenred México. (2023). *Flotilla de autos: ¿cómo administrarla de forma eficiente?*. Recuperado de https://www.edenred.mx/blog/flotilla-de-autos-como-administrarla

- Elmasri, R., & Navathe, S. B. (2017). *Fundamentals of Database Systems* (7th ed.). Pearson.

- MySQL Documentation. (2025). *MySQL 8.0 Reference Manual*. Oracle Corporation.