**MySQL2**

Maria José Bastilla Osorio

Laboratorio

campus lands

Programación

Bucaramanga , santander

2024

# Tabla de contenido

# Introducción

En este documento encontrarás detallados los pasos necesarios para la creación de la base de datos de la empresa Autorental. Este proceso es fundamental para organizar de manera efectiva todas las operaciones relacionadas con la gestión de datos, garantizando así un uso óptimo de la base de datos y proporcionando a la empresa un software integral y completo. Este enfoque no solo busca optimizar la estructura y el rendimiento de la base de datos, sino también asegurar que cada procedimiento esté claramente justificado y documentado.

El proceso se inicia con la extracción y análisis de la información proporcionada por la empresa. Esta información se transforma en diagramas y modelos que facilitan la optimización de datos y procesos. Esta fase inicial es crucial para comprender a fondo las necesidades y requisitos específicos de Autorental, permitiendo así diseñar una base de datos que no solo sea funcional, sino que también esté alineada con los objetivos estratégicos y operativos de la empresa.

Además, durante la implementación de la base de datos, se aplican metodologías y prácticas recomendadas para garantizar la integridad, seguridad y eficiencia de los datos almacenados. Se establecen procedimientos de prueba rigurosos para validar la funcionalidad del software y su capacidad para manejar diferentes escenarios y cargas de trabajo. Este enfoque asegura que la base de datos no solo sea robusta desde el punto de vista técnico, sino que también sea fácilmente mantenible y escalable a medida que Autorental continúa creciendo y expandiéndose.

## Caso de Estudio

Mediante la realización de este caso de estudio se busca resolver la problemática que esta sufriendo Autorental la cual no tiene el programa adecuado para la recolección y manejo de datos de su empresa lo que como consecuencia a tenido perdida de datos historiales de clientes entre otras , es necesario resolver este problema con urgencia ya que esta compañía se encuentra en expansión y sin el debido manejo de datos no podrá tener el control total de la

compañía , La solución recomendada es crear una base de datos donde la compañía tenga la visualización total de cada procedimiento que hace la empresa como lo son la entrada de usuarios y la optimización de detalles de cada servicio ofrecido por la compañía.

# Instalación General

## Requisitos Previos

- Servidor de Base de Datos: MySQL o MariaDB
- Herramienta de Administración de Base de Datos: MySQL Workbench, phpMyAdmin, o cualquier otro cliente SQL.
- Lenguaje de Programación: Se recomienda utilizar Python, Java, PHP o cualquier otro lenguaje compatible con bases de datos SQL.
- Entorno de Desarrollo: Un editor de texto como Visual Studio Code o un IDE como PyCharm, Eclipse, etc.

# Problema – Laboratorio:

La empresa donde usted trabaja ha sido contratada para desarrollar un sistema de información para una empresa de alquiler de vehículos llamada AutoRental, y usted ha sido designado para diseñar una base de datos para ese sistema de información.

AutoRental cuenta con 5 sucursales en diferentes ciudades y se proyecta a expandirse a otras ciudades del país y cuenta con una flota propia de vehículos de diferentes tipos, modelos (año), capacidad, etc.

Los clientes de AutoRental podrán alquilar un vehículo en una sucursal y entregarlo en otra sucursal.

AutoRental ofrece descuentos sobre diferentes tipos de vehículos a lo largo del año. Los valores de alquiler dependen del tipo de vehículo (sedán, compacto, camioneta platón, camioneta lujo, deportivo, etc) y se cobran por días y/o semanas. Por ejemplo, si un alquila un vehículo por 9 días, el valor cotizado será de 1 semana y 2 días.

Si un cliente entrega el vehículo pasado la fecha de entrega contratada, se cobrarán los díasadicionales con un incremento del 8%.

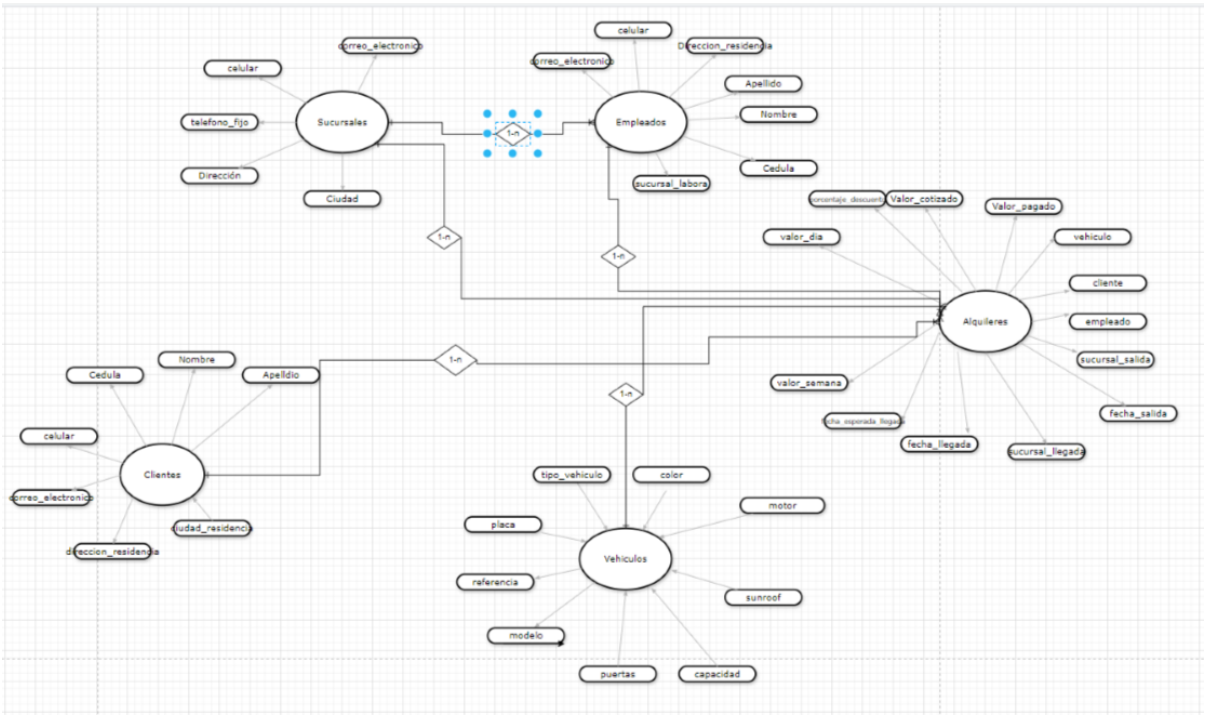
A continuación se va a realizar el proceso para la creación y ejecución de la base de datos :

# Modelo Conceptual

Se construye un diagrama modelo conceptual con los requerimientos propuestos por la compañía:

## Descripción

Modelo conceptual que representa las 5 tablas principales que pide la compañía las cuales son , clientes , vehículos , alquileres , empleados y sucursales , acompañadas de todos los atributos o columnas actualmente necesarias para cada tabla , para finalmente encontrar 5 tipos de relaciones entre ellas .




## Descripción Técnica

Modelo conceptual que contiene 5 nodos como representación de cada tabla , alrededor se visualiza a cada nodo con sus respectivos atributos .
El nodo de sucursales contiene dos tipo de relaciones de tipo (1-n) una con la tabla de empleados y alquileres debido a que dentro de los atributos de el nodo de empleados hay un dato de se recibe del nodo de sucursales y la otra con el nodo de alquileres por un dato en común al igual que las otras relaciones de tipo (1-n) hacia la tabla de alquileres .

Descripción de atributos

- Sucursales
- Ciudad: Dato de tipo texto.
- Dirección:  Dato de tipo texto.
- Teléfono fijo:  Dato de tipo número.
- Celular:  Dato de tipo número.

- Correo electrónico:  Dato de tipo texto.

- Empleados
- Sucursal donde labora:  Dato de tipo texto.
- Cédula:  Dato de tipo número.
- Nombres:  Dato de tipo texto.
- Apellidos:  Dato de tipo texto.
- Dirección de residencia:  Dato de tipo texto.
- Ciudad de residencia:  Dato de tipo texto.
- Celular:  Dato de tipo número.
- Correo electrónico:  Dato de tipo texto.

- Clientes
- Cédula:  Dato de tipo número.
- Nombres:  Dato de tipo texto.
- Apellidos:  Dato de tipo texto.
- Dirección de residencia: Dato de tipo texto.
- Ciudad de residencia:  Dato de tipo texto.
- Celular: Dato de tipo número.
- Correo electrónico:  Dato de tipo texto.

- Vehículos
- Tipo de vehículo:  Dato de tipo texto.
- Placa:  Dato de tipo texto.
- Referencia:  Dato de tipo texto.
- Modelo:  Dato de tipo número.
- Puertas:  Dato de tipo número.
- Capacidad:  Dato de tipo número.
- Sunroof:  Dato de tipo booleano.
- Motor: Dato de tipo texto.
- Color: Dato de tipo texto.

- Alquileres
- Vehículo: Dato de tipo texto.
- Cliente:  Dato de tipo número.
- Empleado:  Dato de tipo número.
- Sucursal de salida:  Dato de tipo texto.
- Fecha de salida:  Dato de tipo fecha.
- Sucursal de llegada:  Dato de tipo texto.
- Fecha de llegada:  Dato de tipo fecha.
- Fecha esperada de llegada: Dato de tipo fecha.
- Valor de alquiler por semana: Dato de tipo número.
- Valor de alquiler por día:  Dato de tipo número.
- Porcentaje de descuento:  Dato de tipo número.
- Valor cotizado:  Dato de tipo número.
- Valor pagado: Dato de tipo número.

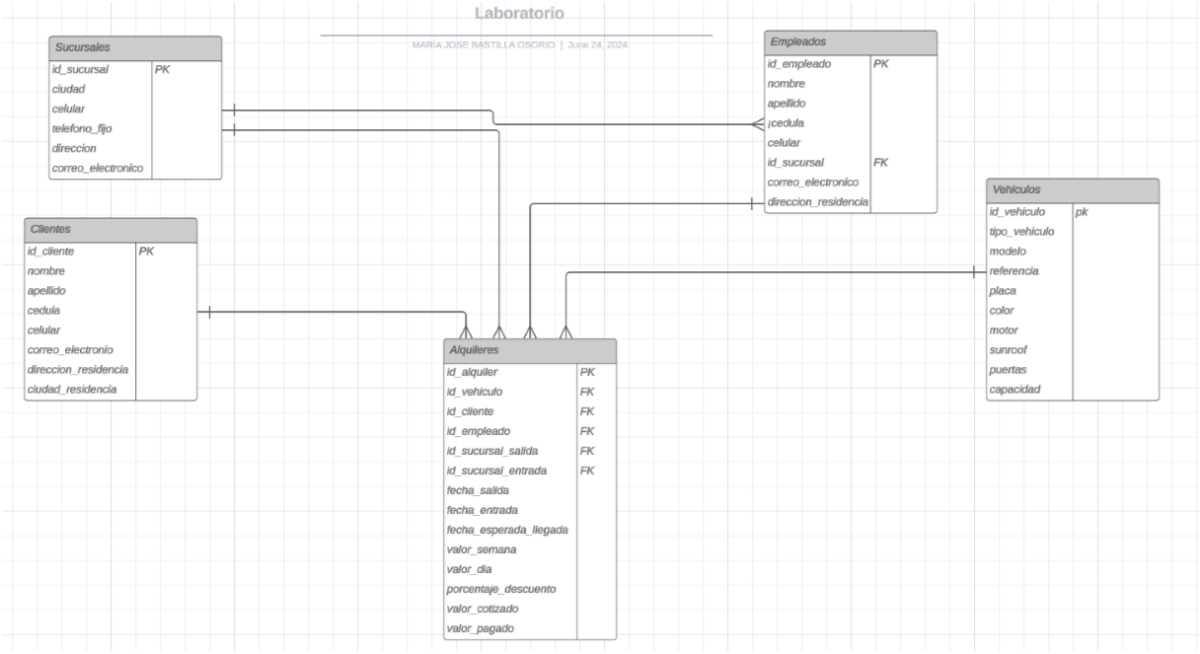
# Modelo Lógico

Se construye un diagrama modelo lógico con los requerimientos propuestos por la compañía:

**Descripción**

Se construye un modelo lógico con la referencia del modelo conceptual , se le añade a cada tabla un id para cada único para cada proceso en cada tabla para llevar mas control y darle un id único a cada proceso se definen cuales son las primary key y las llaves foráneas delante de cada atributo.

**Diseño**




**Descripción Técnica**

Diagrama representa un sistema de gestión de alquiler de vehículos con cinco tablas principales: Sucursales, Clientes, Empleados, Vehículos y Alquileres. Las relaciones entre estas tablas se establecen mediante claves foráneas. La tabla Sucursales contiene información sobre cada sucursal y se relaciona con Empleados y Alquileres a través de id_sucursal. La tabla Clientes almacena datos de los clientes y se relaciona con Alquileres mediante id_cliente. La tabla Empleados, que incluye detalles de los empleados, está vinculada a Sucursales y Alquileres mediante id_sucursal y id_empleado, respectivamente. La tabla Vehículos contiene información de los vehículos disponibles y se relaciona con Alquileres mediante id_vehiculo. Finalmente, la tabla Alquileres registra los detalles de cada alquiler, enlazando con las tablas Sucursales, Clientes, Empleados y Vehículos a través de sus respectivas claves foráneas.

# Normalización

A continuación de normaliza el modelo lógico presentado anteriormente:

## Primera Forma Normal (1FN)

**Descripción**

La primera forma normal (1FN) se enfoca en eliminar los grupos repetitivos dentro de una tabla, asegurando que cada campo contenga solo valores atómicos e indivisibles.

**Descripción Técnica**

En este diagrama, cada campo en las tablas `Sucursales`, `Clientes`, `Empleados`, `Vehículos` y `Alquileres` contiene valores únicos y atómicos, lo que significa que las tablas

están en la primera forma normal. No hay grupos repetitivos ni campos que contengan listas de valores.

## Segunda Forma Normal (2FN)

**Descripción**
La segunda forma normal (2FN) se enfoca en eliminar la dependencia parcial, asegurando que cada campo no clave dependa completamente de la clave primaria.

**Descripción Técnica**
Para que las tablas estén en 2FN, deben estar en 1FN y todos los campos no clave deben depender completamente de la clave primaria. En este diagrama, las tablas `Sucursales`, `Clientes`, `Empleados`, `Vehículos` y `Alquileres` cumplen con esta forma normal, ya que cada campo no clave depende totalmente de su clave primaria respectiva. Por ejemplo, en la tabla `Alquileres`, los campos `fecha_salida`, `fecha_entrada`, `valor_semana`, etc., dependen completamente de la clave primaria `id_alquiler`.

# Tercera Forma Normal (3FN)

**Descripción**
La tercera forma normal (3FN) se enfoca en eliminar la dependencia transitiva, asegurando que todos los campos no clave sean mutuamente independientes.

**Descripción Técnica**
Para que las tablas estén en 3FN, deben estar en 2FN y todos los campos no clave deben ser independientes entre sí. En este diagrama, las tablas están en 3FN. Por ejemplo, en la tabla `Empleados`, los campos `nombre`, `apellido`, `celular`, etc., dependen solo de `id_empleado` y no entre sí. No hay dependencias transitivas presentes, lo que garantiza que las tablas estén correctamente normalizadas en la tercera forma normal.

# Construcción del Modelo Físico

A continuación se crea el modelo lógico de la base de datos :

**Descripción**
Dentro del modelo lógico empezamos creando la base datos para seguidamente crear las tablas mencionadas anteriormente , luego de esto se hace la inserción de los datos que van a estar dentro de la base de datos y finalmente se crean las consultas , los eventos , los triggers y los procedimientos necesarios usando para este proyecto.

**Código**

```
CREATE TABLE sucursales (
  id_sucursal INT PRIMARY KEY,
```

```
    ciudad VARCHAR(100) NOT NULL,
    direccion VARCHAR(200) NOT NULL,
    telefono_fijo VARCHAR(15) NOT NULL,
    correo_electronico VARCHAR(100) NOT NULL
);

CREATE TABLE clientes (
    id_cliente INT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    apellido VARCHAR(100) NOT NULL,
    cedula VARCHAR(15) NOT NULL,
    direccion_residencia VARCHAR(200) NOT NULL,
    ciudad_residencia VARCHAR(100) NOT NULL,
    celular VARCHAR(15) NOT NULL
);

CREATE TABLE vehiculos (
    id_vehiculo INT PRIMARY KEY,
    tipo_vehiculo VARCHAR(50) NOT NULL,
    placa VARCHAR(20) NOT NULL,
    referencia VARCHAR(50) NOT NULL,
    modelo INT NOT NULL,
    puertas INT NOT NULL,
    capacidad INT NOT NULL,
    sunroof BOOLEAN NOT NULL,
    motor VARCHAR(20) NOT NULL,
    color VARCHAR(30) NOT NULL
);
alter table vehiculos modify column placa VARCHAR(30);

CREATE TABLE empleados (
    id_empleado INT PRIMARY KEY,
    id_sucursal INT NOT NULL,
    nombre VARCHAR(100) NOT NULL,
    apellido VARCHAR(100) NOT NULL,
    cedula VARCHAR(15) NOT NULL,
    direccion_residencia VARCHAR(200) NOT NULL,
    correo_electronico VARCHAR(100) NOT NULL,
    celular VARCHAR(15) NOT NULL,
    FOREIGN KEY (id_sucursal) REFERENCES sucursales(id_sucursal)
);

CREATE TABLE alquileres (
    id_alquiler INT PRIMARY KEY,
    id_vehiculo INT NOT NULL,
    id_cliente INT NOT NULL,
    id_empleado INT NOT NULL,
```

```
    id_sucursal_salida INT NOT NULL,
    id_sucursal_entrada INT NOT NULL,
    fecha_salida DATE NOT NULL,
    fecha_entrada DATE NOT NULL,
    fecha_esperada_llegada DATE NOT NULL,
    valor_semana DECIMAL(10,2) NOT NULL,
    valor_dia DECIMAL(10,2) NOT NULL,
    porcentaje_descuento DECIMAL(5,2) NOT NULL,
    valor_cotizado DECIMAL(10,2) NOT NULL,
    valor_pagado DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (id_vehiculo) REFERENCES vehiculos(id_vehiculo),
    FOREIGN KEY (id_cliente) REFERENCES clientes(id_cliente),
    FOREIGN KEY (id_empleado) REFERENCES empleados(id_empleado),
    FOREIGN KEY (id_sucursal_salida) REFERENCES sucursales(id_sucursal),
    FOREIGN KEY (id_sucursal_entrada) REFERENCES sucursales(id_sucursal)
);
```

## Descripción Técnica

El script SQL define cinco tablas para un sistema de alquiler de vehículos: `sucursales` para los detalles de las sucursales, `clientes` para la información de los clientes, `vehiculos` para los datos de los vehículos, `empleados` para los datos de los empleados, y `alquileres` para gestionar los registros de alquileres. Las tablas están interrelacionadas mediante claves foráneas que conectan vehículos, clientes, empleados y sucursales, asegurando integridad referencial y registrando detalles clave de cada transacción de alquiler.

# Tablas

## Tabla sucursales

### Descripción

Se crea tabla de sucursales para almacenar información crucial como lo son, id único para identificar, ciudad de permanecía, dirección, teléfono fijo y correo electrónico

### Gráfica

### Descripción Técnica

Tabla de sucursales con la información de las 5 sucursales pertenecientes a la compañía , con una relación de (1-n) con la tabla de alquileres ya que comparte el dato de su id para informar acerca de las sucursales a donde llega y entra cada automóvil al momento d prestar un servicio a algún cliente.

## Tabla clientes

### Descripción

La tabla de clientes es creada para llevar un registro de las personas que han tomado o están interesados en tomar algún servicio con la empresa esto con el fin de llevar un control total de estos, esta contiene datos como: id único para cada cliente, nombre y apellido para tener un segundo identificar, su cedula como segundo identificador único, dirección de residencia, ciudad de residencia y celular.

### Gráfica

### Descripción Técnica

Se ha diseñado la tabla "clientes" con el propósito de mantener un registro exhaustivo de individuos que han utilizado o muestran interés en los servicios proporcionados por la empresa. Esta estructura de datos permite un control detallado de dichos clientes , la tabla mantiene una relación de (1-n) con la tabla de alquileres proporcionándole su id a dicha tabla para tener la información acerca de quien toma el servicio.

## Tabla vehículos

### Descripción

La tabla de vehículos es creada para llevar un control total de los vehículos pertenecientes a la empresa, esta tabla brinda información la cual podrá mirar el cliente para elegir su vehículo y la empresa para llevar el control y cambios de cada vehículo, tiene datos esenciales como : id unico para cada vehículo , tipo de vehículo (sedán, compacto, camioneta platón, camioneta lujo, deportivo, etc) , placa, referencia, modelo, puertas, capacidad de personas, sunroof , motor y color.

### Gráfica

### Descripción Técnica

Se ha desarrollado la tabla "vehículos" con el objetivo de gestionar de manera integral los vehículos pertenecientes a la empresa. Esta estructura de datos facilita tanto a los clientes la visualización y selección de vehículos disponibles como a la empresa el control y mantenimiento de cada unidad , la tabla mantiene una relacion de (1-n) con la tabla de alquileres proporcionándole su id a dicha tabla para tener la informacion acerca de que automóvil esta tomado para el servicio.

## Tabla empleados

### Descripción

Se crea tabla de empleados para registrar cada empleado perteneciente a la compañía con el fin de llevar el control de las personas con diferentes permisos dentro de la base de datos de la empresa contiene atributos como: id único para cada empleado, id de la sucursal donde labora id obtenida de la tabla de sucursales, nombre, apellido, cedula, dirección de residencia, correo electrónico y celular.

### Gráfica

### Descripción Técnica

Se ha implementado la tabla "empleados" con el propósito de gestionar el registro detallado de cada empleado que forma parte de la compañía. Esta estructura de datos permite el control eficiente de las personas con diferentes niveles de permisos dentro del sistema de gestión de la empresa ,  la tabla mantiene una relación de (1-n) con la tabla de alquileres proporcionándole su id a dicha tabla para tener la información acerca de el empleado encargado del servicio.

## Tabla alquileres

### Descripción

Tabla de clientes se crea para llevar control total del servicio que se ofrece por la compañía que es el alquiler de autos dentro de esta tabla podemos encontrar datos del id del vehículo usado , id del cliente que va a usar el servicio , el id del empleado encargado del servicio , id de la sucursal de donde sale el carro , id de la sucursal donde llega el carro y finalmente datos detallados acerca del servicio obtenido por algún cliente ( Se tiene que tener en cuenta que el id se llama de las tablas creadas con anterioridad para tener control de datos de las personas involucradas a este ).

**Gráfica**

**Descripción Técnica**

Se ha desarrollado la tabla "servicios_alquiler_autos" para gestionar integralmente el servicio de alquiler de autos ofrecido por la compañía. Esta estructura de datos permite un control exhaustivo de cada transacción de alquiler , Es importante destacar que los IDs referidos son obtenidos de las tablas previamente creadas para asegurar un seguimiento efectivo de todos los datos relacionados con las personas involucradas en cada transacción.

# Relaciones entre Tablas

**Relaciones entre Tablas**

**Descripción**

El diagrama muestra las relaciones entre las tablas de una base de datos de un sistema de alquiler de vehículos. Las tablas principales son "Sucursales", "Clientes", "Empleados", "Vehículos" y "Alquileres". Las relaciones se establecen principalmente a través de claves foráneas que conectan las tablas relacionadas.

**Descripción Técnica**

1. **Sucursales** (id_sucursal, ciudad, celular, telefono_fijo, direccion, correo_electronico)
    - Relacionada con **Empleados** mediante id_sucursal (clave foránea en Empleados).
    - Relacionada con **Alquileres** mediante id_sucursal_salida y id_sucursal_entrada (claves foráneas en Alquileres).
2. **Clientes** (id_cliente, nombre, apellido, cedula, celular, correo_electronico, direccion_residencia, ciudad_residencia)
    - Relacionada con **Alquileres** mediante id_cliente (clave foránea en Alquileres).
3. **Empleados** (id_empleado, nombre, apellido, cedula, celular, id_sucursal, correo_electronico, direccion_residencia)
    - Relacionada con **Sucursales** mediante id_sucursal (clave foránea en Empleados).
    - Relacionada con **Alquileres** mediante id_empleado (clave foránea en Alquileres).
4. **Vehículos** (id_vehiculo, tipo_vehiculo, modelo, referencia, placa, color, motor, sunroof, puertas, capacidad)

- Relacionada con **Alquileres** mediante id_vehiculo (clave foránea en Alquileres).

5. **Alquileres** (id_alquiler, id_vehiculo, id_cliente, id_empleado, id_sucursal_salida, id_sucursal_entrada, fecha_salida, fecha_entrada, fecha_esperada_llegada, valor_semana, valor_dia, porcentaje_descuento, valor_cotizado, valor_pagado)

    - Relacionada con **Clientes** mediante id_cliente.
    - Relacionada con **Empleados** mediante id_empleado.
    - Relacionada con **Sucursales** mediante id_sucursal_salida y id_sucursal_entrada.
    - Relacionada con **Vehículos** mediante id_vehiculo.

# Inserción de Datos

### *Descripción*

**Sucursales:** Se establecen 5 inserciones con los datos de las 5 sucursales que hay en el pais con el siguiente formato.

**Clientes:** Se establecen 100 inserciones con los datos de los clientes que ha tenido la empresa con el siguiente formato.

**Empleados**: Se establecen 100 inserciones con los datos de los empleados actuales en la empresa con el siguiente formato.

**Vehículos:** Se establecen 100 inserciones con los datos de los vehiculos vinculados con la empresa para el servicio que se ofrece de autorentar con el siguiente formato u orden.

**Alquileres:** Se establecen 100 inserciones con los datos detallados de cada servicio que ha sido comprado y usado con el siguiente formato u orden.

### *Gráfica*

```
INSERT INTO sucursales (id_sucursal, ciudad, direccion, telefono_fijo, correo_electronico) VALUES
(1, 'Bogotá', 'Calle 123 #45-67', '6011234567', 'bogota@sucursal.com'),
```

```
INSERT INTO clientes (id_cliente, nombre, apellido, cedula, direccion_residencia, ciudad_residencia, celular)
VALUES
(1, 'Pedro', 'Ramírez', '6789012345', 'Calle 16 #17-18', 'Bogotá', '3001234567'),
```

```
INSERT INTO vehiculos (id_vehiculo, tipo_vehiculo, placa, referencia, modelo, puertas, capacidad, sunroof, motor, color) VALUES
(1, 'Sedán', 'ABC123', 'Toyota Corolla', 2022, 4, 5, TRUE, '1.8L', 'Rojo'),
```

```
INSERT INTO empleados (id_empleado, id_sucursal, nombre, apellido, cedula, direccion_residencia, correo_electronico, celular) VALUES
(1, 1, 'Pedro', 'Ramírez', '6789012345', 'Calle 16 #17-18', 'pedro.ramirez@autorental.com', '3101234567'),
```

```
INSERT INTO alquileres (id_alquiler, id_vehiculo, id_cliente, id_empleado, id_sucursal_salida, id_sucursal_entrada, fecha_salida, fecha_entrada,
VALUES
(1, 1, 1, 1, 1, 2, '2024-06-01', '2024-06-08', '2024-06-07', 700.00, 100.00, 10.00, 800.00, 720.00),
```

**Descripción Técnica**

**sucursales** (id_sucursal, ciudad, direccion, telefono_fijo, correo_electronico).

**clientes** (id_cliente, nombre, apellido, cedula, direccion_residencia, ciudad_residencia, celular) .

**empleados** (id_empleado, id_sucursal, nombre, apellido, cedula, direccion_residencia, correo_electronico, celular).

**vehiculos** (id_vehículo, tipo_vehículo, placa, referencia, modelo, puertas, capacidad, sunroof, motor, color).

**alquileres** (id_alquiler, id_vehículo, id_cliente, id_empleado, id_sucursal_salida, id_sucursal_entrada, fecha_salida, fecha_entrada, fecha_esperada_llegada, valor_semana, valor_dia, porcentaje_descuento, valor_cotizado, valor_pagado).

# Funciones y Procedimientos

A continuacion podra visualizar las funciones y procedimientos dentro de la base de datos :

**Funcion#1**

**Descripción:**

Esta función calcula el valor cotizado de un alquiler basado en el tipo de vehículo y la cantidad de días de alquiler.

**Sintaxis**

CREATE FUNCTION calcular_valor_cotizado(tipo_vehiculo VARCHAR(50), dias INT)
RETURNS DECIMAL(10,2)
```
```

**Parámetros**

- `tipo_vehiculo`: Tipo de vehículo (VARCHAR(50)).
- `dias`: Número de días de alquiler (INT).

**Salida/Retorno:**

- `DECIMAL(10,2)`: Valor cotizado del alquiler.

**Ejemplo de Implementación:**

```
SELECT calcular_valor_cotizado('Sedán', 10);
```

**Funcion#2**

**Descripción**

Esta función calcula el valor pagado de un alquiler considerando posibles retrasos en la devolución.

**Sintaxis**

```
CREATE FUNCTION calcular_valor_pagado(fecha_devolucion DATE,
fecha_esperada_llegada DATE, valor_cotizado DECIMAL(10,2), valor_dia
DECIMAL(10,2))
RETURNS DECIMAL(10,2)
```

**Parámetros**

- `fecha_devolucion`: Fecha de devolución del vehículo (DATE).
- `fecha_esperada_llegada`: Fecha esperada de devolución del vehículo (DATE).
- `valor_cotizado`: Valor cotizado del alquiler (DECIMAL(10,2)).
- `valor_dia`: Valor diario del alquiler (DECIMAL(10,2)).

**Salida/Retorno:**

- `DECIMAL(10,2)`: Valor pagado del alquiler considerando retrasos.

**Ejemplo de Implementación**

```
SELECT calcular_valor_pagado('2024-06-15', '2024-06-10', 1000.00, 50.00);
```

## Procedimientos

### Procedimiento#1

### Descripción

Un procedimiento almacenado para registrar un nuevo alquiler en la base de datos.

### Sintaxis

```
 DELIMITER //

CREATE PROCEDURE registrar_alquiler(
    IN p_id_alquiler INT,
    IN p_id_vehiculo INT,
    IN p_id_cliente INT,
    IN p_id_empleado INT,
    IN p_id_sucursal_salida INT,
    IN p_id_sucursal_entrada INT,
    IN p_fecha_salida DATE,
    IN p_fecha_entrada DATE,
    IN p_fecha_esperada_llegada DATE,
    IN p_valor_semana DECIMAL(10,2),
    IN p_valor_dia DECIMAL(10,2),
    IN p_porcentaje_descuento DECIMAL(5,2),
    IN p_valor_cotizado DECIMAL(10,2),
    IN p_valor_pagado DECIMAL(10,2)
)
BEGIN
    INSERT INTO alquileres (
        id_alquiler, id_vehiculo, id_cliente, id_empleado, id_sucursal_salida,
        id_sucursal_entrada, fecha_salida, fecha_entrada, fecha_esperada_llegada,
        valor_semana, valor_dia, porcentaje_descuento, valor_cotizado, valor_pagado
    )
    VALUES (
        p_id_alquiler, p_id_vehiculo, p_id_cliente, p_id_empleado, p_id_sucursal_salida,
        p_id_sucursal_entrada, p_fecha_salida, p_fecha_entrada, p_fecha_esperada_llegada,
        p_valor_semana, p_valor_dia, p_porcentaje_descuento, p_valor_cotizado,
p_valor_pagado
    );
END //

DELIMITER ;
```

**Parámetros**

p_id_alquiler: ID del alquiler (INT).
p_id_vehiculo: ID del vehículo (INT).
p_id_cliente: ID del cliente (INT).
p_id_empleado: ID del empleado (INT).
p_id_sucursal_salida: ID de la sucursal de salida (INT).
p_id_sucursal_entrada: ID de la sucursal de entrada (INT).
p_fecha_salida: Fecha de salida (DATE).
p_fecha_entrada: Fecha de entrada (DATE).
p_fecha_esperada_llegada: Fecha esperada de llegada (DATE).
p_valor_semana: Valor semanal del alquiler (DECIMAL).
p_valor_dia: Valor diario del alquiler (DECIMAL).
p_porcentaje_descuento: Porcentaje de descuento aplicado (DECIMAL).
p_valor_cotizado: Valor cotizado para el alquiler (DECIMAL).
p_valor_pagado: Valor pagado por el cliente (DECIMAL).

**Salida/Retorno**

Inserción de un nuevo registro en la tabla alquileres.

**Ejemplo de Implementación**

```
 CALL registrar_alquiler(
    1, 101, 201, 301, 1, 2, '2024-07-01', '2024-07-07', '2024-07-07',
    700.00, 100.00, 10.00, 650.00, 650.00
);
```

## Triggers

**Trigger#1**

**Descripción**

Este trigger verifica que el modelo de un vehículo no sea anterior a 2012 antes de insertar o actualizar un registro en la tabla `vehiculos`.

**Sintaxis**

```
CREATE TRIGGER trigger_check_modelo_before
BEFORE INSERT OR UPDATE ON vehiculos
```

**Parámetros**

- Ninguno.

**Salida/Retorno:**

- Ninguno.

**Ejemplo de Implementación**

```
CREATE TRIGGER trigger_check_modelo_before
BEFORE INSERT
ON vehiculos FOR EACH ROW
BEGIN
    IF NEW.modelo < 2012 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'El modelo no puede ser anterior a 2012.';
    END IF;
END;
```

**Trigger2**

**Descripción**

Este trigger guarda el cambio de número de celular de los clientes en una tabla de logs cada vez que se actualiza el campo `celular` en la tabla `clientes`.

**Sintaxis**

```
CREATE TRIGGER trigger_guardar_celular_after_update
AFTER UPDATE ON clientes
```

**Parámetros**

- Ninguno.

**Salida/Retorno**

- Ninguno.

**Ejemplo de Implementación**

```
CREATE TRIGGER trigger_guardar_celular_after_update
```

```
AFTER UPDATE ON clientes
FOR EACH ROW
BEGIN
   IF OLD.celular <> NEW.celular THEN
      INSERT INTO log_cambios_celular (id_cliente, old_celular, new_celular)
      VALUES (OLD.id_cliente, OLD.celular, NEW.celular);
   END IF;
END;
```

## Eventos

### Eventos#1

#### Descripción

Un evento programado para actualizar el estado de los alquileres vencidos diariamente.

#### Sintaxis

```
DELIMITER //

CREATE EVENT actualizar_estado_alquileres_vencidos
ON SCHEDULE EVERY 1 DAY
STARTS '2024-07-01 00:00:00'
DO
BEGIN
   UPDATE alquileres
   SET valor_pagado = valor_pagado + (DATEDIFF(CURDATE(), fecha_esperada_llegada)
* valor_dia * 1.08)
   WHERE fecha_esperada_llegada < CURDATE() AND fecha_entrada IS NULL;
END //

DELIMITER ;
```

#### Parámetros

- Ninguno.

#### Salida/Retorno

Actualización del campo valor_pagado para alquileres vencidos.

#### Ejemplo de Implementación

El evento se ejecutará automáticamente todos los días a medianoche.

# Consultas

## Consultas de Cliente

### Consulta# 1

### Descripción

Esta consulta obtiene los vehículos tipo 'Sedán' disponibles para alquiler en junio de 2024, con un valor semanal entre 500 y 1000.

### Sintaxis

```
SELECT v.id_vehiculo, v.tipo_vehiculo, v.placa, v.referencia, v.modelo, v.color, a.valor_semana, a.valor_dia
FROM vehiculos v
LEFT JOIN alquileres a ON v.id_vehiculo = a.id_vehiculo
WHERE (a.fecha_salida IS NULL OR a.fecha_entrada < CURDATE())
AND v.tipo_vehiculo = 'Sedán'
AND a.valor_semana BETWEEN 500 AND 1000
AND CURDATE() BETWEEN '2024-06-01' AND '2024-06-30';
```

### Parámetros

- Ninguno.

### Salida/Retorno

- Una lista de vehículos tipo 'Sedán' disponibles para alquiler.

### Ejemplo de Implementación

```
SELECT v.id_vehiculo, v.tipo_vehiculo, v.placa, v.referencia, v.modelo, v.color, a.valor_semana, a.valor_dia
FROM vehiculos v
LEFT JOIN alquileres a ON v.id_vehiculo = a.id_vehiculo
WHERE (a.fecha_salida IS NULL OR a.fecha_entrada < CURDATE())
AND v.tipo_vehiculo = 'Sedán'
AND a.valor_semana BETWEEN 500 AND 1000
AND CURDATE() BETWEEN '2024-06-01' AND '2024-06-30';
```

**Consulta# 2**

**Descripción**

Esta consulta muestra todos los registros de la tabla `alquileres`.

**-Sintaxis:**

```
SELECT * FROM alquileres;
```

**Parámetro**

- Ninguno.

**Salida/Retorno**

- Todos los registros de la tabla `alquileres`.

**Ejemplo de Implementación**

```
SELECT * FROM alquileres;
```

**Consulta #3**

**Descripción**

Esta consulta obtiene el historial de alquileres de un cliente específico (con `id_cliente` 91).

**-Sintaxis**

```
SELECT id_alquiler, fecha_entrada, fecha_salida, valor_pagado
FROM alquileres
WHERE id_cliente = 91;
```

**Parámetros**

- `id_cliente`: ID del cliente (INT).

**Salida/Retorno:**

- Historial de alquileres del cliente con ID 91.

**Ejemplo de Implementación:**

```
SELECT id_alquiler, fecha_entrada, fecha_salida, valor_pagado
FROM alquileres
WHERE id_cliente = 91;
```

**Consulta #4**

**Descripción**

Esta consulta permite a los clientes ver todos los vehículos disponibles ordenados por referencia.

**Sintaxis**

```
SELECT * FROM vehiculos ORDER BY referencia;
```

### Parámetros
- Ninguno.

### Salida/Retorno

- Lista de vehículos ordenados por referencia.

### Ejemplo de Implementación
```
SELECT * FROM vehiculos ORDER BY referencia;
```

## Consulta #5

### Descripción
Esta consulta permite a los clientes ver su historial de alquileres con el total pagado y fechas correspondientes

### Sintaxis
```
SELECT a.*, SUM(valor_pagado) AS total_pagado
FROM alquileres a
WHERE a.id_cliente = :id_cliente
GROUP BY a.id_alquiler, a.fecha_salida, a.fecha_entrada, a.fecha_esperada_llegada,
    a.valor_semana, a.valor_dia, a.porcentaje_descuento, a.valor_cotizado;
```

### Parámetros
- `id_cliente`: ID del cliente (INT).

### Salida/Retorno
- Historial de alquileres con total pagado y fechas.

### -Ejemplo de Implementación
```
SELECT a.*, SUM(valor_pagado) AS total_pagado
FROM alquileres a
WHERE a.id_cliente = 101
GROUP BY a.id_alquiler, a.fecha_salida, a.fecha_entrada, a.fecha_esperada_llegada,
    a.valor_semana, a.valor_dia, a.porcentaje_descuento, a.valor_cotizado;
```

## Consulta# 6

### Descripción
Esta consulta permite a los clientes ver las especificaciones de un vehículo con el nombre del carro.

### Sintaxis
```
SELECT * FROM vehiculos WHERE referencia = :nombre_carro;
```

**Parámetros**

- `nombre_carro`: Nombre del carro (VARCHAR).

**Salida/Retorno**

- Especificaciones del vehículo con el nombre dado.

**Ejemplo de Implementación**

```
SELECT * FROM vehiculos WHERE referencia = 'Toyota Corolla';
```

**Consulta# 7**

**Descripción**

Esta consulta permite a los clientes ver los empleados disponibles en una sucursal específica para solicitar ayuda.

**Sintaxis**

```
SELECT e.id_empleado, e.nombre, e.apellido, e.correo_electronico, e.celular
FROM empleados e
WHERE e.id_sucursal = :id_sucursal;
```

**Parámetros:**

- `id_sucursal`: ID de la sucursal (INT).

**Salida/Retorno**

- Lista de empleados disponibles en la sucursal dada.

**Ejemplo de Implementación**

```
SELECT e.id_empleado, e.nombre, e.apellido, e.correo_electronico, e.celular
FROM empleados e
WHERE e.id_sucursal = 1;
```

## Consultas de Empleado

**Consulta# 1**

**Descripción**

Esta consulta permite a los empleados ver el historial de empleados por sucursal.

**Sintaxis**

```
SELECT e.*
FROM empleados e
WHERE e.id_sucursal = :id_sucursal;
```

### Parámetros

- `id_sucursal`: ID de la sucursal (INT).

### Salida/Retorno

- Historial de empleados por sucursal.

### Ejemplo de Implementación

```
SELECT e.*
FROM empleados e
WHERE e.id_sucursal = 2;
```

## Consulta #2

### Descripción:

Esta consulta permite a los empleados ver el historial de alquileres por sucursal.

### Sintaxis

```
SELECT a.*
FROM alquileres a
WHERE a.id_sucursal_salida = :id_sucursal
   OR a.id_sucursal_entrada = :id_sucursal;
```

### Parámetros

- `id_sucursal`: ID de la sucursal (INT).

### Salida/Retorno

- Historial de alquileres por sucursal.

### Ejemplo de Implementación

```
SELECT a.*
FROM alquileres a
WHERE a.id_sucursal_salida = 2
   OR a.id_sucursal_entrada = 2;
```

## Consulta #3

### Descripción

Esta consulta permite a los empleados ver todos los vehículos ordenados alfabéticamente por referencia.

**-Sintaxis**

SELECT * FROM vehiculos ORDER BY referencia;

**Parámetros:**

- Ninguno.

**Salida/Retorno**

- Lista de vehículos ordenados alfabéticamente.

**Ejemplo de Implementación:**

SELECT * FROM vehiculos ORDER BY referencia;

## Consulta# 4

**-Descripción**

Esta consulta permite a los empleados ver las veces que un empleado ha estado involucrado en un alquiler

**Sintaxis**

```
SELECT e.id_empleado, e.nombre, e.apellido, COUNT(a.id_alquiler) AS num_alquileres
FROM empleados e
JOIN alquileres a ON e.id_empleado = a.id_empleado
GROUP BY e.id_empleado, e.nombre, e.apellido;
```

- Ninguno.

**Salida/Retorno**

- Número de alquileres en los que ha estado involucrado cada empleado.

**-Ejemplo de Implementación**

```
SELECT e.id_empleado, e.nombre, e.apellido, COUNT(a.id_alquiler) AS num_alquileres
FROM empleados e
JOIN alquileres a ON e.id_empleado = a.id_empleado
GROUP BY e.id_empleado, e.nombre, e.apellido;
```

## Consulta# 5

**Descripción**

Esta consulta permite a los empleados ver los alquileres en los que se pagó menos de lo cotizado.

**Sintaxis**

```
SELECT *
FROM alquileres
WHERE valor_pagado < valor_cotizado;
```

**Parámetros:**

- Ning

## Usuarios y Accesos

## Tabla de Usuarios

| Usuario | Acceso | Descripción | Comandos | Funciones/Procedimientos / consultas |
|---|---|---|---|---|
| Cliente (web-sofware) | % | Este usuario fue creados para que los clientes pudieran entrar a la plataforma y hacer cierto tipo de consultas sin entrar o tener control total de los datos de la compañia | GRANT SELECT ON | Consultas clientes |
| Empleado | % | Este usuario fue creado para que los empleado (administracion , generencia , empleados ) puedan tener control para seleccionar , visualizar insertar y actualizar datos acerca de todas las tablas pero nunca para eliminar . | GRANT SELECT, INSERT, UPDATE | funcion1(), Funcion2(), Consultas empleados |

# Referencias