

**LABORATORIO AUTORENTA**

**ZULLY FERNANDA ORTIZ AVENDAÑO**

**DOCENTE**

**PEDRO FELIPE GÓMEZ BONILLA**

**CAMPUSLANDS  
SANDBOX  
RUTA JAVA  
TIBU  
2024**

<b>Introducción</b>	<b>4</b>
<b>Caso de Estudio</b>	<b>5</b>
<b>Planificación</b>	<b>6</b>
Ejecución	6
Construcción del Modelo Conceptual	6
Descripción	6
Las entidades y atributos	6
Relaciones y Cardinalidades	8
Gráfica	9
Construcción del Modelo Lógico	10
Descripción	10
Relaciones y Cardinalidades	12
Gráfica	13
Normalización del Modelo Lógico	14
Primera Forma Normal (1FN)	14
Descripción	14
Descripción técnica	14
Gráfica	15
Forma Normal (2FN)	16
Descripción	16
Descripción técnica	16
Gráfica	17
Tercera Forma Normal (3FN)	18
Descripción	18
Descripción técnica	18
Gráfica	19
Construcción del Modelo Físico	20
Descripción	20
Construcción del Diagrama UML	23
Descripción	23
<b>Inserciones de Datos</b>	<b>27</b>
Descripción	27
Consultas de Datos	31
<b>Procedimientos</b>	<b>37</b>
1. Procedimiento: InsertarSucursal	37
2. Procedimiento: InsertarEmpleado	38
3. Procedimiento: InsertarVehiculo	39
4. Procedimiento: InsertarTipoVehiculo	40
5. Procedimiento: InsertarClientes	41
6. Procedimiento: ActualizarCliente	42
7. Procedimiento: EliminarDescuento	43

8. Procedimiento: ListarSucursal	44
9. Procedimiento: descuento	45
10. Procedimiento: AplicarDescuento	46

# Introducción

En este proyecto se presentará una guía detallada de la estructura e implementación de una base de datos para la empresa de alquiler de vehículos llamada AutoRenta, donde el objetivo principal es gestionar un sistema de información.

Primero se realizará un caso de estudio con sus requerimientos específicos, donde a partir de estos análisis se procederá a desarrollar un modelo conceptual detallado donde se identificará las entidades principales, sus atributos y relaciones de todas ellas, donde este paso es realizado para comprender la estructura que se llevará a cabo para la empresa AutoRenta.

Prosiguiendo se realizará la conversión del modelo conceptual al lógico, donde este tiene como función representar de manera más concisa de la organización de toda la base de datos, facilitando si compression, también se aplicará el proceso de normalización hasta la tercera forma normal (3FN) donde se reducira los datos redundantes.

Finalmente se llevará a cabo la conversión del modelo lógico al modelo físico, el cual definirá las entidades, atributos y relaciones.

Con estos pasos se garantizara un proyecto completo y efectivo para el diseño de los datos de datos para la empresa AutoRenta.

# Caso de Estudio

La empresa Autorenta ha solicitado el desarrollo de un sistema de información para gestionar su operación de alquiler de vehículos. Actualmente, con 5 sucursales en distintas ciudades y planes de expansión, requiere un sistema para gestionar el alquiler de su flota de vehículos. Los clientes podrán recoger un vehículo en una sucursal y devolverlo en otra. Las tarifas se calculan por días y/o semanas según el tipo de vehículo, con posibles descuentos durante el año. Si se excede la fecha de entrega, se aplicará un recargo del 8% por cada día adicional. El sistema busca optimizar la gestión de inventarios, reservas y pagos, mejorando la eficiencia y satisfacción del cliente, por lo que tendrá los siguientes requerimientos para la elaboración:

- *Sucursales:*

ciudad y dirección donde se ubica, teléfono fijo, celular y correo electrónico.

- *Empleados:*

sucursal donde labora, cédula, nombres, apellidos, dirección y ciudad de residencia, celular y correo electrónico.

- *Clientes:*

cédula, nombres, apellidos, dirección y ciudad de residencia, celular y correo Electrónico.

- *Vehículos:*

tipo de vehículo, placa, referencia, modelo, puertas, capacidad, sunroof, motor, Color.

- *Alquileres:*

vehículo, cliente, empleado, sucursal y fecha de salida, sucursal y fecha de llegada, fecha esperada de llegada, valor de alquiler por semana, valor de alquiler por día, porcentaje de descuento, valor cotizado y valor pagado.

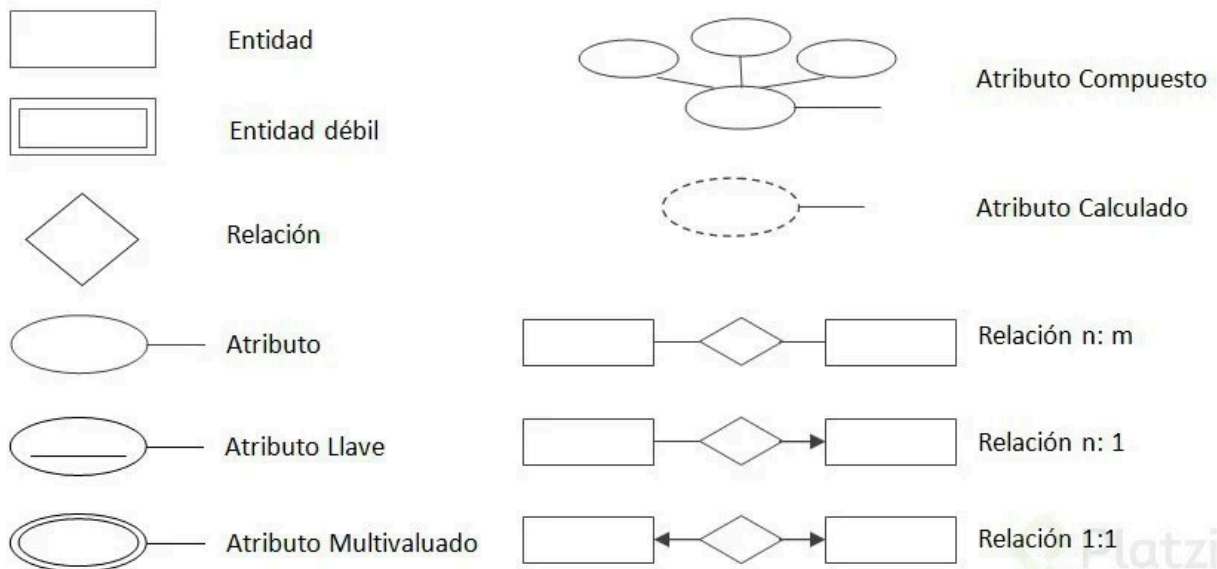
# Planificación

## Ejecución

Una vez haya sido analizado toda la información requerida para la elaboración de base de datos de la empresa Autorenta, se inició la elaboración del modelo conceptual, la cual este proceso proporciona un alta descripción donde representará los conceptos principales de la base de datos y las relaciones que hay entre ellos.

## Construcción del Modelo Conceptual

Se realizó el Modelo conceptual para identificar cada una de las entidades, sus atributos y las relaciones entre ellas, donde se proporcionó una visión más clara y estructurada de cómo se organizó y conecto los diferentes modelos de la base de dato teniendo en cuenta lo siguiente:



## Descripción

### Las entidades y atributos

#### 1. Sucursales:

- ❖ Id\_sucursal.
- ❖ Ubicación.
- ❖ Teléfono fijo.
- ❖ Celular.

- ❖ Correo Electronico.

## 2. Empleados:

- ❖ Id\_empleado.
- ❖ id\_sucursal.
- ❖ Cédula.
- ❖ Nombres.
- ❖ Apellidos.
- ❖ Ubicación.
- ❖ Celular.
- ❖ Correo electrónico.

## 3. Vehículos:

- ❖ Id\_vehiculo.
- ❖ Tipo vehiculo.
- ❖ Placa.
- ❖ Referencia.
- ❖ Modelo.
- ❖ Puertas.
- ❖ Capacidad.
- ❖ Sunroof.
- ❖ Motor.
- ❖ Color.

## 4. Clientes:

- ❖ Id\_cliente.
- ❖ Cédula.
- ❖ Nombres.
- ❖ Apellidos.
- ❖ Ubicación.
- ❖ Celular.
- ❖ Correo electrónico.

## 5. Alquileres:

- ❖ Id\_alquiler.
- ❖ Id\_vehiculo.
- ❖ Id\_empleado.
- ❖ Id\_sucursal.
- ❖ Fecha salida
- ❖ Fecha esperada entrega.

- ❖ Fecha entrega.
- ❖ Valor cotizado.
- ❖ Valor pagado.

6. tipo\_vehiculo:

- ❖ Id\_tipo\_vehiculo.
- ❖ Tipo
- ❖ Valor\_alquiler\_dia.
- ❖ Valor\_alquiler\_semana.

7. Descuento:

- ❖ Id\_descuento.
- ❖ Id\_tipo\_vehiculo
- ❖ Fecha\_inicio
- ❖ Fecha\_fin
- ❖ porcentaje \_descuento.

## Relaciones y Cardinalidades

1. Sucursales - Empleados:

- ❖ Relación: “Tiene”, Un sucursal tiene varios empleados y un empleado tiene una sucursal.
- ❖ Cardinalidad: 1-N (uno a muchos).

2. Empleados - Alquileres:

- ❖ Relación: “Adquirir”, Un empleado puede adquirir varios alquileres y un alquiler puede ser adquirido por un empleado.
- ❖ Cardinalidad: 1:N (uno a muchos).

3. Vehículo - Alquileres:

- ❖ Relación: ‘Alquilar’, Un vehículo puede ser alquilado varias veces y en cada alquiler se puede alquilar un vehículo.
- ❖ Cardinalidad: 1:N (uno a muchos).

4. Tipo\_vehiculo - vehiculo:

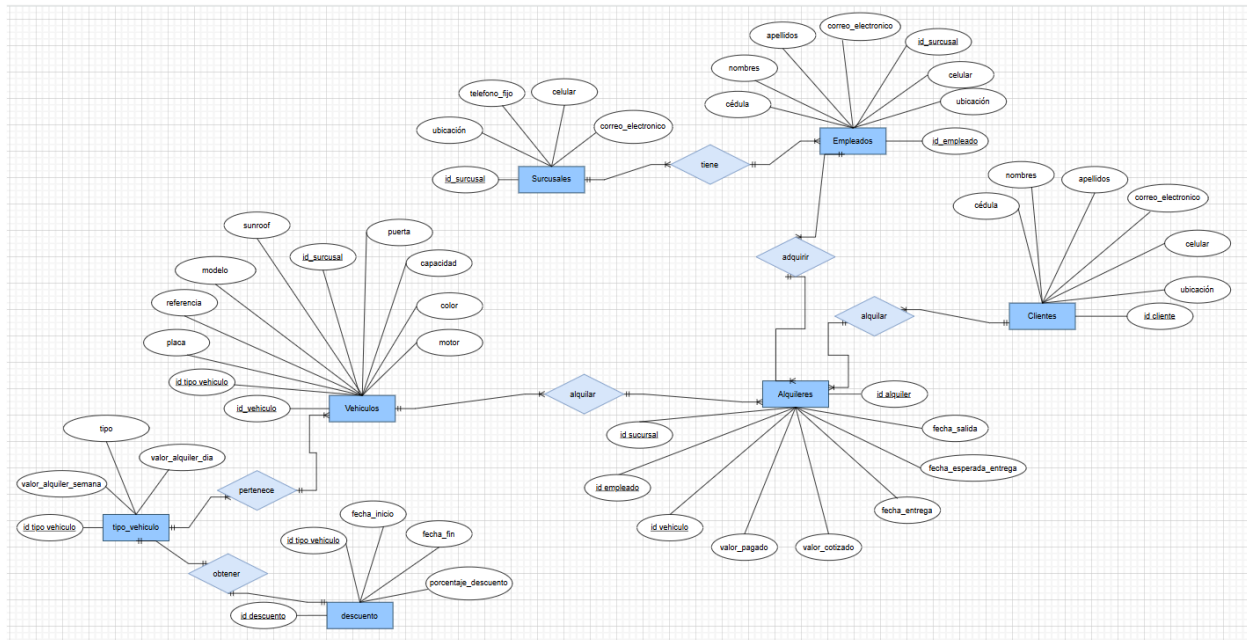
- ❖ Relación: ‘Pertenece’, Muchos vehículos le pertenecen a un tipo de vehículo y un tipo vehículo le pertenece a muchos vehículos.
- ❖ Cardinalidad: 1:N (Uno a muchos).

5. Tipo\_vehiculo - descuento:



- ❖ Relación: 'Obtener', Un tipo de vehículo puede tener un descuento y un descuento está relacionado con un tipo de vehículo.
  - ❖ Cardinalidad: 1:1 (Uno a uno ).
6. Clientes - Alquileres:
- ❖ Relación: 'Realizar', Un cliente puede realizar varios alquileres pero un alquiler solo lo puede hacer un cliente.
  - ❖ Cardinalidad: 1:N (Uno a Muchos ).

## Gráfica



# Construcción del Modelo Lógico

Se realizó el modelo lógico teniendo en cuenta el modelo conceptual, donde se incorporaron detalles más específicos como las características de cada atributo, incluidas las claves primarias, foráneas y las relaciones de cardinalidad.

## Descripción

### 1. Sucursales:

- ❖ Id\_sucursal int primary key.
- ❖ Ubicación varchar(50).
- ❖ Teléfono fijo int.
- ❖ Celular int.
- ❖ Correo Electronico varchar(50).

### 2. Empleados:

- ❖ Id\_empleado int primary key.
- ❖ Id\_sucursal foreign key.
- ❖ Cédula int.
- ❖ Nombres varchar(50).
- ❖ Apellidos varchar(50).
- ❖ Ubicación varchar(50).
- ❖ Celular int.
- ❖ Correo electrónico varchar(50).

### 3. Vehículos:

- ❖ Id\_vehiculo primary key.
- ❖ Tipo vehículo varchar(50).
- ❖ Placa varchar(50).
- ❖ Referencia int.
- ❖ Modelo varchar(50).
- ❖ Puertas int.
- ❖ Capacidad int.
- ❖ Sunroof varchar(50).
- ❖ Motor varchar(50).
- ❖ Color varchar(50).

4. Clientes:

- ❖ Id\_cliente primary key.
- ❖ Cédula int.
- ❖ Nombres varchar(50).
- ❖ Apellidos varchar(50).
- ❖ Ubicación varchar(50).
- ❖ Celular int.
- ❖ Correo electrónico varchar(50).

5. Alquileres:

- ❖ Id\_alquiler Primary key.
- ❖ Id\_vehiculo foreign key.
- ❖ Id\_empleado foreign key.
- ❖ Id\_sucursal foreign key.
- ❖ Fecha salida Date
- ❖ Fecha esperada entrega date.
- ❖ Fecha entrega Date
- ❖ Valor cotizado int.
- ❖ Valor pagado int.

6. tipo\_vehiculo:

- ❖ Id\_tipo\_vehiculo Primary key.
- ❖ Tipo varchar(50).
- ❖ Valor\_alquiler\_dia Date.
- ❖ Valor\_alquiler\_semana Date.

7. Descuento:

- ❖ Id\_descuento Primary key.
- ❖ Id\_tipo\_vehiculo
- ❖ Fecha\_inicio Date.
- ❖ Fecha\_fin Date.
- ❖ porcentaje \_descuento double.

## Relaciones y Cardinalidades

Se realizó las relaciones y cardinalidades respectivas del modelo lógico con sus entidades para tener mejor visualización de la base de datos:

### 1. Sucursales - Empleados:

- ❖ Una sucursal tiene varios empleados y un empleado tiene una sucursal. 1-N (uno a muchos).



### 2. Empleados - Alquileres:

- ❖ Un empleado puede adquirir varios alquileres y un alquiler puede ser adquirido por un empleado. 1:N (uno a muchos).



### 3. Vehículo - Alquileres:

- ❖ Un vehículo puede ser alquilado varias veces y en cada alquiler se puede alquilar un vehículo. 1:N (uno a muchos).



### 4. Tipo\_vehiculo - vehiculo:

- ❖ Muchos vehículos le pertenece a un tipo de vehiculo y un tipo vehiculo le pertenece a muchos vehículos. 1:N (Uno a muchos).



### 5. Tipo\_vehiculo - descuento:

- ❖ Un tipo de vehículo puede tener un descuento y un descuento está relacionado con un tipo de vehículo. 1:1 (Uno a uno ).



### 6. Clientes - Alquileres:

- ❖ Un cliente puede realizar varios alquileres pero un alquiler solo lo puede hacer un cliente. 1:N (Uno a muchos).



## Gráfica



# Normalización del Modelo Lógico

Se realizó el proceso de la normalización de las tablas anteriormente visualizadas para organizar los datos de manera más eficiente, minimizando redundancias y dependencias transitivas en la base de datos en desarrollo.

## Primera Forma Normal (1FN)

Una tabla está en 1FN si cumple con los siguientes criterios:

- Cada atributo debe tener valores atómicos.
- Cada fila la misma tabla debe ser única
- Debe prevalecer un crecimiento vertical de los datos y no horizontal
- No deben existir grupos repetidos de datos.

### Descripción

La primera forma normal es el primer nivel de normalización en base de datos donde se le aplicará a las tablas de las bases de datos para garantizar una mejor organización donde se evita redundancias y asegurará la consistencia de la información.

### Descripción técnica

1. Sucursales:
  - ❖ No encuentra en 1FN, ya que cada columna no tiene valores únicos y son repetitivos pero cuenta con una clave primaria única.
2. Empleados :
  - ❖ No encuentra en 1FN, ya que cada columna no tiene valores únicos y son repetitivos pero cuenta con una clave primaria única.
3. Clientes :
  - ❖ No encuentra en 1FN, ya que cada columna no tiene valores únicos y son repetitivos pero cuenta con una clave primaria única.
4. Alquileres :
  - ❖ No encuentra en 1FN, ya que cada columna no tiene valores únicos y son repetitivos pero cuenta con una clave primaria única.
5. Vehiculos:
  - ❖ No encuentra en 1FN, ya que cada columna no tiene valores únicos y son repetitivos pero cuenta con una clave primaria única.

6. Descuentos:

- ❖ No encuentra en 1FN, ya que cada columna no tiene valores únicos y son repetitivos pero cuenta con una clave primaria única.

7. Tipo\_vehiculo:

- ❖ No encuentra en 1FN, ya que cada columna no tiene valores únicos y son repetitivos pero cuenta con una clave primaria única.

## Gráfica

Sucursales						
id_sucursal	titulo	ciudad	dirección	telefono_fijo	Celular	correo_electronico
PK						

Empleados										
id_empleado	id_sucursal	cedula	nombre1	nombre2	apellido1	apellido2	ciudad	ubicación	celular	correo_electronico
PK	FK									

Clientes									
id_cliente	cedula	nombre1	nombre2	apellido1	apellido2	ciudad	ubicación	celular	correo_electronico
PK									

Alquileres								
id_alquiler	id_sucursal	id_empleado	id_vehiculo	valor_pagado	valor_cotizado	fecha_entrega	fecha_esperada_entrega	fecha_salida
PK	FK	FK	FK					

Vehiculos										
id_vehiculo	id_tipo_vehiculo	id_sucursal	placa	modelo	sunroof	referencia	puerta	capacidad	color	motor
PK	FK	FK								

Descuento				
id_descuento	id_tipo_vehiculo	fecha_inicio	fecha_fin	porcentaje_descuento
PK	FK			

Tipo_vehiculo			
id_tipo_vehiculo	valor_alquiler_semanal	valor_alquiler_dia	tipo
PK			

## Forma Normal (2FN)

Una tabla está en 2FN si cumple con los siguientes criterios:

- Estar en 1FN.
- La relación debe tener una clave principal, de preferencia simple.
- Cada atributo de la tabla debe depender totalmente del atributo clave.

### Descripción

La segunda forma normal, es el segundo nivel de normalización en el diseño de la base de datos que se aplicará a las tablas de una base de datos que ya cumplen con la primera forma normal y lleva a cabo la eliminación de dependencias parciales dentro de una tabla.

### Descripción técnica

1. Sucursales :
  - ❖ Se encuentra en 2FN, ya que cuenta con una clave primaria única y cada La columna depende completamente de esa clave primaria.
2. Empleados :
  - ❖ Se encuentra en 2FN, ya que cuenta con una clave primaria única y cada La columna depende completamente de esa clave primaria.
3. Clientes :
  - ❖ Se encuentra en 2FN, ya que cuenta con una clave primaria única y cada La columna depende completamente de esa clave primaria.
4. Alquileres :
  - ❖ Se encuentra en 2FN, ya que cuenta con una clave primaria única y cada La columna depende completamente de esa clave primaria.
5. Vehiculos:
  - ❖ Se encuentra en 2FN, ya que cuenta con una clave primaria única y cada



La columna depende completamente de esa clave primaria.

6. Descuento:

❖ Se encuentra en 2FN, ya que cuenta con una clave primaria única y cada La columna depende completamente de esa clave primaria.

7. Tipo\_vehiculo:

❖ Se encuentra en 2FN, ya que cuenta con una clave primaria única y cada La columna depende completamente de esa clave primaria.

## Gráfica

Sucursales						
id_sucursal	titulo	ciudad	dirección	telefono_fijo	Celular	correo_electronico
PK						

Empleados										
id_empleado	id_sucursal	cedula	nombre1	nombre2	apellido1	apellido2	ciudad	ubicación	celular	correo_electronico
PK	FK									

Clientes									
id_cliente	cedula	nombre1	nombre2	apellido1	apellido2	ciudad	ubicación	celular	correo_electronico
PK									

Alquileres									
id_alquiler	id_sucursal	id_empleado	id_vehiculo	valor_pagado	valor_cotizado	fecha_entrega	fecha_esperada_entrega	fecha_salida	
PK	FK	FK	FK						

Vehiculos										
id_vehiculo	id_tipo_vehiculo	id_sucursal	placa	modelo	sunroof	referencia	puerta	capacidad	color	motor
PK	FK	FK								

Descuento				
id_descuento	id_tipo_vehiculo	fecha_inicio	fecha_fin	porcentaje_descuento
PK	FK			

Tipo_vehiculo			
id_tipo_vehiculo	valor_alquiler_semanal	valor_alquiler_dia	tipo
PK			

## Tercera Forma Normal (3FN)

Una tabla está en 3NF si cumple con los siguientes criterios:

- Debe estar en 2FN.
- No deben existir atributos no principales que dependen transitivamente del atributo clave.

### Descripción

La tercera forma normal, es el tercer nivel de normalización en el diseño de la base de datos que se aplicará a las tablas de una base de datos que ya cumplen con la segunda forma normal y se enfoca en la eliminación de dependencias transitivas, evitando que un atributo no clave depende de otro no clave.

### Descripción técnica

1. Sucursales :
  - ❖ Se encuentra en 3FN, ya que está en la 2FN y en cada columna no hay dependencias transitivas con la clave primaria.
2. Empleados :
  - ❖ Se encuentra en 3FN, ya que está en la 2FN y en cada columna no hay dependencias transitivas con la clave primaria.
3. Clientes :
  - ❖ Se encuentra en 3FN, ya que está en la 2FN y en cada columna no hay dependencias transitivas con la clave primaria.
4. Alquileres :
  - ❖ Se encuentra en 3FN, ya que está en la 2FN y en cada columna no hay dependencias transitivas con la clave primaria.

5. Vehiculos:
  - ❖ Se encuentra en 3FN, ya que está en la 2FN y en cada columna no hay dependencias transitivas con la clave primaria.
6. Descuento:
  - ❖ Se encuentra en 3FN, ya que está en la 2FN y en cada columna no hay dependencias transitivas con la clave primaria.
7. Tipo\_vehiculo:
  - ❖ Se encuentra en 3FN, ya que está en la 2FN y en cada columna no hay dependencias transitivas con la clave primaria.

## Gráfica

Sucursales						
id_sucursal	titulo	ciudad	dirección	telefono_fijo	Celular	correo_electronico
PK						

Empleados										
id_empleado	id_sucursal	cedula	nombre1	nombre2	apellido1	apellido2	ciudad	ubicación	celular	correo_electronico
PK	FK									

Clientes									
id_cliente	cedula	nombre1	nombre2	apellido1	apellido2	ciudad	ubicación	celular	correo_electronico
PK									

Alquileres								
id_alquiler	id_sucursal	id_empleado	id_vehiculo	valor_pagado	valor_cotizado	fecha_entrega	fecha_esperada_entrega	fecha_salida
PK	FK	FK	FK					

Vehiculos										
id_vehiculo	id_tipo_vehiculo	id_sucursal	placa	modelo	sunroof	referencia	puerta	capacidad	color	motor
PK	FK	FK								

Descuento				
id_descuento	id_tipo_vehiculo	fecha_inicio	fecha_fin	porcentaje_descuento
PK	FK			

Tipo_vehiculo			
id_tipo_vehiculo	valor_alquiler_semanal	valor_alquiler_dia	tipo
PK			

## Construcción del Modelo Físico

Se realizó el modelo físico acorde con la elaboración del modelo lógico donde incluye las entidades, sus atributos y las relaciones entre ellas. El objetivo de este modelo es incorporar los tipos de datos de los atributos previamente definidos.

### Descripción

El modelo físico se trabaja en MySQL, en el cual cada entidad se representa como una tabla compuesta por sus atributos correspondientes, organizados en columnas con tipos de datos específicos según sea necesario

### Tablas

Para crear la base de datos utilice el siguiente comando:

```
create database AutoRenta_OrtizZully;
```

Para utilizar la base de datos ocupe el siguiente comando:

```
use AutoRenta_OrtizZully;
```

Comenzaremos creando las tablas junto con sus tipos de datos correspondientes. Para esto, utiliza los siguientes comandos:

#### 1. Creación de la tabla Sucursales:

```
CREATE TABLE Sucursales (
  id_sucursal INT PRIMARY KEY,
  ubicacion varchar(50),
  direccion VARCHAR(50) ,
```

```
ciudad VARCHAR(50) ,  
telefono_fijo VARCHAR(50) ,  
celular VARCHAR(50) ,  
correo_electronico VARCHAR(255)  
);
```

2. Creación de la tabla tipo\_vehiculo:

```
CREATE TABLE Tipo_vehiculo (  
    id_tipoV INT PRIMARY KEY,  
    valor_alquiler_semana INT,  
    valor_alquiler_dia INT,  
    tipo VARCHAR(55)  
);
```

3. Creación de la tabla vehiculos:

```
CREATE TABLE Vehiculos (  
    id_vehiculo INT PRIMARY KEY,  
    placa VARCHAR(50),  
    referencia INT,  
    modelo VARCHAR(50),  
    puertas INT,  
    capacidad INT,  
    sunroof VARCHAR(30),  
    motor VARCHAR(50),  
    color VARCHAR(40),  
    id_tipoV INT,  
    FOREIGN KEY (id_tipoV) REFERENCES Tipo_vehiculo(id_tipoV)  
);
```

4. Creación de la tabla descuento:

```
CREATE TABLE Descuentos (  
    id_descuento INT PRIMARY KEY,  
    fecha_inicio DATE,  
    fecha_fin DATE,  
    porcentaje_descuento INT,  
    id_tipoV INT,
```

```
FOREIGN KEY (id_tipoV) REFERENCES Tipo_vehiculo(id_tipoV)
);
```

5. Creación de la tabla clientes:

```
CREATE TABLE Clientes (
id_cliente INT PRIMARY KEY,
cedula VARCHAR(50),
nombre1 VARCHAR(60),
nombre2 VARCHAR(60),
apellido1 VARCHAR(60),
apellido2 VARCHAR(60),
direccion VARCHAR(50),
ciudad VARCHAR(50),
celular VARCHAR(20),
correo_electronico VARCHAR(50)
);
```

6. Creación de la tabla empleados:

```
CREATE TABLE Empleados (
id_empleado INT PRIMARY KEY,
cedula VARCHAR(50),
nombre1 VARCHAR(60),
nombre2 VARCHAR(60),
apellido1 VARCHAR(60),
apellido2 VARCHAR(60),
direccion VARCHAR(50),
ciudad VARCHAR(50),
celular VARCHAR(50),
correo_electronico VARCHAR(50),
id_sucursal INT,
FOREIGN KEY (id_sucursal) REFERENCES Sucursales(id_sucursal)
);
```

7. Creación de la tabla alquileres:

```
CREATE TABLE Alquileres (
id_alquiler INT PRIMARY KEY,
fecha_salida DATE,
```

```
fecha_llegada VARCHAR(55) NULL,  
fecha_esperada DATE,  
valor_cotizado INT,  
valor_pagado INT,  
id_vehiculo INT,  
id_cliente INT,  
id_empleado INT,  
id_sucursal INT,  
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) REFERENCES Sucursales(id_sucursal)  
);
```

## Construcción del Diagrama UML

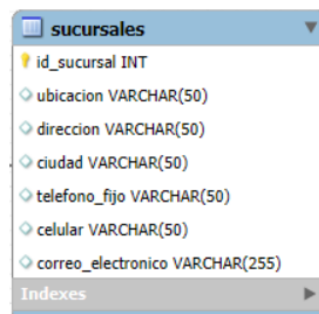
Se realizó un diagrama UML tomando como referencia la normalización para poder entender mejor cada diseño, la arquitectura del código y la implementación. Dónde nos permitirá tener una visión clara y detallada de cómo se maneja cada consulta de la la empresa AutoRenta.

### Descripción

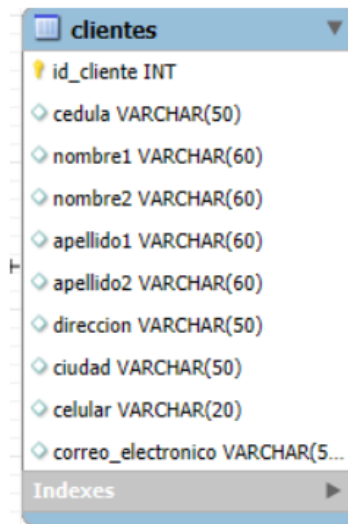
El diagrama UML se diseñó con el objetivo de representar detalladamente la estructura de cada tabla y sus relaciones, donde tendrá así más ganadas unas llaves primarias y foráneas, además especifica obligatoriamente los atributos, dando un nivel de detalle facilitada para poder entenderse con más facilidad.

Comenzaremos creando las tablas junto con sus tipos de datos correspondientes:

#### 1.Tabla Sucursales:



2. Tabla Clientes:



The screenshot shows the 'clientes' table structure. It has a primary key 'id\_cliente' of type INT. Other fields include 'cedula' (VARCHAR(50)), 'nombre1' (VARCHAR(60)), 'nombre2' (VARCHAR(60)), 'apellido1' (VARCHAR(60)), 'apellido2' (VARCHAR(60)), 'direccion' (VARCHAR(50)), 'ciudad' (VARCHAR(50)), 'celular' (VARCHAR(20)), and 'correo\_electronico' (VARCHAR(50)). An 'Indexes' tab is visible at the bottom.

Field	Type
id_cliente	INT
cedula	VARCHAR(50)
nombre1	VARCHAR(60)
nombre2	VARCHAR(60)
apellido1	VARCHAR(60)
apellido2	VARCHAR(60)
direccion	VARCHAR(50)
ciudad	VARCHAR(50)
celular	VARCHAR(20)
correo_electronico	VARCHAR(50)

3. Tabla Empleados:



The screenshot shows the 'empleados' table structure. It has a primary key 'id\_empleado' of type INT. Other fields include 'cedula' (VARCHAR(50)), 'nombre1' (VARCHAR(60)), 'nombre2' (VARCHAR(60)), 'apellido1' (VARCHAR(60)), 'apellido2' (VARCHAR(60)), 'direccion' (VARCHAR(50)), 'ciudad' (VARCHAR(50)), 'celular' (VARCHAR(50)), 'correo\_electronico' (VARCHAR(50)), and a foreign key 'id\_sucursal' (INT). An 'Indexes' tab is visible at the bottom.

Field	Type
id_empleado	INT
cedula	VARCHAR(50)
nombre1	VARCHAR(60)
nombre2	VARCHAR(60)
apellido1	VARCHAR(60)
apellido2	VARCHAR(60)
direccion	VARCHAR(50)
ciudad	VARCHAR(50)
celular	VARCHAR(50)
correo_electronico	VARCHAR(50)
id_sucursal	INT

4. Tabla Alquileres:



alquileres	
id_alquiler	INT
fecha_salida	DATE
fecha_llegada	VARCHAR(55)
fecha_esperada	DATE
valor_cotizado	INT
valor_pagado	INT
id_vehiculo	INT
id_cliente	INT
id_empleado	INT
id_sucursal	INT
Indexes	

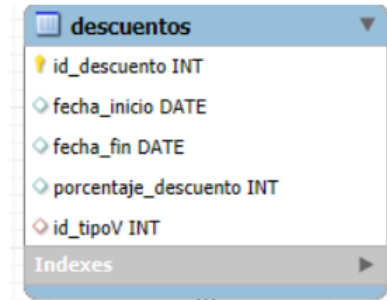
##### 5. Tabla Vehículos:

vehiculos	
id_vehiculo	INT
placa	VARCHAR(50)
referencia	INT
modelo	VARCHAR(50)
puertas	INT
capacidad	INT
sunroof	VARCHAR(30)
motor	VARCHAR(50)
color	VARCHAR(40)
id_tipoV	INT
Indexes	

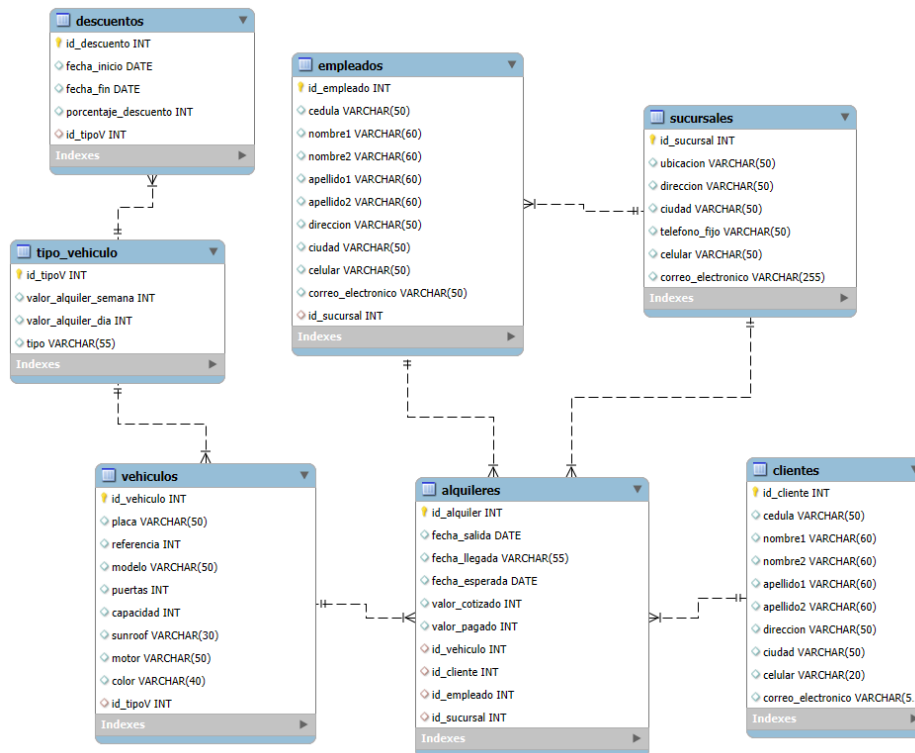
##### 6. Tabla tipo\_vehiculo

tipo_vehiculo	
id_tipoV	INT
valor_alquiler_semana	INT
valor_alquiler_dia	INT
tipo	VARCHAR(55)
Indexes	

##### 7. Tabla tipo descuento:



## Gráfica



# Inserciones de Datos

## Descripción

Para poder entender mejor el proceso de inserción de datos en la tabla de datos de AutoRental, se deberá insertar datos en todos los datos creados

### 1. Tabla Sucursales:

```
INSERT INTO Sucursales (id_sucursal, ubicacion, direccion, ciudad, telefono_fijo, celular, correo_electronico) VALUES
(1, 'Centro Comercial Plaza', 'Bogotá', 'Carrera 15 #45-78', '6012345678', '3001234567', 'contacto1@empresa.com'),
(2, 'Avenida Central', 'Medellín', 'Calle 10 #25-30', '6048765432', '3102345678', 'contacto2@empresa.com'),
(3, 'Parque Industrial', 'Cali', 'Avenida 5 #12-25', '6028765432', '3203456789', 'contacto3@empresa.com'),
(4, 'Zona Franca', 'Barranquilla', 'Calle 7 #18-45', '6051234567', '3214567890', 'contacto4@empresa.com'),
(5, 'Centro Histórico', 'Cartagena', 'Carrera 4 #6-10', '6052345678', '3225678901', 'contacto5@empresa.com');
```

### 2. Tabla de tipo\_vehiculo:

```
INSERT INTO Tipo_vehiculo (id_tipoV, valor_alquiler_semana, valor_alquiler_dia, tipo) VALUES
(1, 200000, 30000, 'Sedán'),
(2, 250000, 35000, 'Camioneta'),
(3, 300000, 45000, 'SUV'),
(4, 150000, 25000, 'Compacto'),
(5, 500000, 70000, 'Camioneta de lujo'),
(6, 400000, 55000, 'Deportivo'),
(7, 200000, 32000, 'Minivan'),
(8, 600000, 85000, 'Camioneta platón'),
(9, 350000, 50000, 'Convertible'),
(10, 100000, 15000, 'Híbrido');
```

### 3. Tabla vehículos:

**INSERT INTO** Vehiculos (id\_vehiculo, placa, referencia, modelo, puertas, capacidad, sunroof, motor, color, id\_tipoV)**VALUES**

(1, 'ABC123', 2023, 'Model A', 4, 5, 'Sí', 'V6', 'Rojo', 1),  
(2, 'DEF456', 2023, 'Model B', 4, 5, 'No', 'V8', 'Azul', 2),  
(3, 'GHI789', 2023, 'Model C', 4, 5, 'Sí', 'V6', 'Negro', 3),  
(4, 'JKL012', 2023, 'Model D', 2, 4, 'No', 'V4', 'Blanco', 4),  
(5, 'MNO345', 2023, 'Model E', 4, 5, 'Sí', 'V8', 'Plata', 5),  
(6, 'PQR678', 2022, 'Model F', 4, 5, 'No', 'V6', 'Gris', 1),  
(7, 'STU901', 2022, 'Model G', 2, 2, 'Sí', 'V4', 'Verde', 2),  
(8, 'VWX234', 2022, 'Model H', 4, 7, 'Sí', 'V8', 'Azul', 3),  
(9, 'YZA567', 2021, 'Model I', 4, 5, 'No', 'V6', 'Negro', 4),  
(10, 'BCD890', 2021, 'Model J', 4, 5, 'Sí', 'V8', 'Rojo', 5),  
(11, 'EFG123', 2023, 'Model K', 4, 5, 'No', 'V6', 'Blanco', 1),  
(12, 'HIJ456', 2023, 'Model L', 4, 5, 'Sí', 'V4', 'Azul', 2),  
(13, 'KLM789', 2022, 'Model M', 2, 4, 'No', 'V8', 'Gris', 3),  
(14, 'NOP012', 2022, 'Model N', 4, 5, 'Sí', 'V6', 'Negro', 4),  
(15, 'QRS345', 2021, 'Model O', 4, 7, 'Sí', 'V4', 'Verde', 5),  
(16, 'TUV678', 2020, 'Model P', 4, 5, 'No', 'V8', 'Plata', 1),  
(17, 'WXY901', 2020, 'Model Q', 4, 5, 'Sí', 'V6', 'Rojo', 2),  
(18, 'ZAB234', 2023, 'Model R', 4, 5, 'No', 'V8', 'Negro', 3),  
(19, 'CDE567', 2023, 'Model S', 2, 4, 'Sí', 'V6', 'Blanco', 4),  
(20, 'FGH890', 2022, 'Model T', 4, 5, 'Sí', 'V4', 'Azul', 5);

### 4. Tabla descuento:

**INSERT INTO** Descuentos (id\_descuento, fecha\_inicio, fecha\_fin, porcentaje\_descuento, id\_tipoV) **VALUES**

(1, '2024-11-01', '2024-11-15', 10, 1),  
(2, '2024-11-01', '2024-11-15', 15, 2),  
(3, '2024-11-01', '2024-11-15', 5, 3),  
(4, '2024-11-01', '2024-11-15', 12, 4),  
(5, '2024-11-01', '2024-11-15', 20, 5),  
(6, '2024-11-16', '2024-11-30', 8, 1),  
(7, '2024-11-16', '2024-11-30', 18, 2),  
(8, '2024-11-16', '2024-11-30', 7, 3),  
(9, '2024-11-16', '2024-11-30', 15, 4),  
(10, '2024-11-16', '2024-11-30', 25, 5),

## 5. Tabla clientes:

```
INSERT INTO Clientes (id_cliente, cedula, nombre1, nombre2, apellido1, apellido2, direccion, ciudad, celular, correo_electronico) VALUES
(1, '10987654321', 'Carlos', 'Antonio', 'Sánchez', 'Martínez', 'Calle 1', 'Bogotá', 3001234501, 'carlos.sanchez@cliente.com'),
(2, '20987654321', 'Ana', 'María', 'González', 'Pérez', 'Calle 2', 'Bogotá', 3001234502, 'ana.gonzalez@cliente.com'),
(3, '30987654321', 'Jorge', 'Luis', 'Mendoza', 'Ramírez', 'Calle 3', 'Medellín', 3001234503, 'jorge.mendoza@cliente.com'),
(4, '40987654321', 'Luisa', 'Fernanda', 'Hernández', 'Gómez', 'Carrera 4', 'Medellín', 3001234504, 'luisa.hernandez@cliente.com'),
(5, '50987654321', 'Pedro', 'José', 'Rodríguez', 'Sánchez', 'Avenida 5', 'Cali', 3001234505, 'pedro.rodriguez@cliente.com'),
(6, '60987654321', 'Marta', 'Lucía', 'Vega', 'González', 'Calle 6', 'Cali', 3001234506, 'marta.vega@cliente.com'),
(7, '70987654321', 'Tomás', 'Carlos', 'Ramírez', 'Torres', 'Carrera 7', 'Cartagena', 3001234507, 'tomas.ramirez@cliente.com'),
(8, '80987654321', 'Valentina', 'Andrea', 'Pérez', 'Morales', 'Avenida 8', 'Cartagena', 3001234508, 'valentina.perez@cliente.com'),
(9, '90987654321', 'Ricardo', 'Álvaro', 'Castillo', 'López', 'Calle 9', 'Barranquilla', 3001234509, 'ricardo.castillo@cliente.com'),
(10, '100987654321', 'Elena', 'Sofía', 'Vásquez', 'González', 'Carrera 10', 'Barranquilla', 3001234510, 'elena.vasquez@cliente.com'),
```

## 6. Tabla empleados:

```
INSERT INTO Empleados (id_empleado, cedula, nombre1, nombre2, apellido1, apellido2, direccion, ciudad, celular, correo_electronico, id_sucursal) VALUES
(1, '1234567890', 'Juan', 'Carlos', 'Pérez', 'Gómez', 'Calle 50', 'Bogotá', '3001112233', 'juan@autorental.com', 1),
(2, '1234567891', 'María', 'Luisa', 'Rodríguez', 'Martínez', 'Calle 51', 'Medellín', '3001112234', 'maria@autorental.com', 2),
(3, '1234567892', 'Pedro', 'Antonio', 'García', 'López', 'Carrera 12', 'Cali', '3001112235', 'pedro@autorental.com', 3),
(4, '1234567893', 'Laura', 'Beatriz', 'Martínez', 'Jiménez', 'Avenida 78', 'Barranquilla', '3001112236', 'laura@autorental.com', 4),
(5, '1234567894', 'Carlos', 'Eduardo', 'Lopez', 'Fernández', 'Calle 29', 'Cartagena', '3001112237', 'carlos@autorental.com', 5),
(6, '1234567895', 'Juan', 'Carlos', 'Pérez', 'Gómez', 'Calle 60', 'Bogotá', '3001112238', 'juan.carlos@autorental.com', 1),
(7, '1234567896', 'Alejandra', 'Sofía', 'Martínez', 'Paredes', 'Calle 61', 'Bogotá', '3001112239', 'alejandra@autorental.com', 1),
```

```
(8, '1234567897', 'Carlos', 'Eduardo', 'Méndez', 'Jiménez', 'Calle 62', 'Bogotá', '3001112240',  
'carlos@autorental.com', 1),  
(9, '1234567898', 'Laura', 'Isabel', 'González', 'López', 'Calle 63', 'Bogotá', '3001112241',  
'laura@autorental.com', 1),  
(10, '1234567899', 'Marcos', 'Antonio', 'Lopez', 'Rodríguez', 'Calle 64', 'Bogotá', '3001112242',  
'marcos@autorental.com', 1),
```

## 7. Tabla Alquileres:

```
INSERT INTO Alquileres (id_alquiler, fecha_salida, fecha_llegada, fecha_esperada,  
valor_cotizado, valor_pagado, id_vehiculo, id_cliente, id_empleado, id_sucursal)  
VALUES
```

```
(1, '2024-11-01', '2024-11-05', '2024-11-05', 100000, 100000, 1, 1, 1, 1),  
(2, '2024-11-02', '2024-11-06', '2024-11-06', 150000, 150000, 2, 2, 2, 2),  
(3, '2024-11-03', '2024-11-07', '2024-11-07', 200000, 200000, 3, 3, 3, 3),  
(4, '2024-11-04', '2024-11-08', '2024-11-08', 120000, 120000, 4, 4, 4, 4),  
(5, '2024-11-05', '2024-11-09', '2024-11-09', 180000, 180000, 5, 5, 5, 5),  
(6, '2024-01-05', '2024-01-12', '2024-01-10', 700000, 750000, 6, 6, 3, 2),  
(7, '2024-01-07', '2024-01-14', '2024-01-13', 650000, 650000, 7, 7, 1, 1),  
(8, '2024-01-08', NULL, '2024-01-15', 850000, NULL, 8, 8, 5, 3),  
(9, '2024-01-10', '2024-01-18', '2024-01-17', 950000, 980000, 9, 9, 2, 4),  
(10, '2024-01-11', '2024-01-17', '2024-01-16', 600000, 650000, 10, 10, 4, 2),
```

## Consultas de Datos

Las consultas en una base de datos son indispensables, ya que facilitan el acceso y la Recuperación de información almacenada. Además, permiten mantener la base de datos actualizada mediante la inserción, modificación y actualización de datos. Son fundamentales para almacenar, manipular y recuperar datos de manera eficiente y segura. Para realizar consultas básicas, se utiliza la siguiente sintaxis:

- 1. En esta consulta se muestran todos los datos 'SELECT \*' de una tabla en específico 'FROM clientes' :

```
SELECT * FROM Clientes;
```

- 2. Consultar vehículos disponibles de un tipo específico:

```
SELECT * FROM Vehiculos WHERE id_tipoV = 3;
```

- 3. Listar vehiculos por su numeros de puertas.

```
SELECT * FROM Vehiculos order by puertas asc;
```

- 4. Buscar cliente por cédula.

```
SELECT * FROM Clientes WHERE cedula = '10987654321';
```

- 5. Listar todos los vehículos con sunroof.

```
SELECT * FROM Vehiculos WHERE sunroof = 'Sí';
```

- 6. Contar cuántos vehículos hay por cada tipo.

```
SELECT id_tipoV, COUNT(*) AS cantidad FROM Vehiculos GROUP BY id_tipoV;
```

- 7. Listar empleados de una sucursal

```
SELECT * FROM empleados where id_sucursal = 1;
```

- 8. Listar los descuentos activos en una fecha dada.

```
SELECT * FROM Descuentos WHERE '2024-11-18' BETWEEN fecha_inicio AND fecha_fin;
```

- 9. Listar vehículos alquilados en los últimos 30 días.

```
SELECT * FROM Alquileres WHERE fecha_esperada >= DATE_SUB(CURDATE(),  
INTERVAL 30 DAY);
```

-- 10. Listar vehículos por color.

```
SELECT * FROM Vehiculos WHERE color = 'Rojo';
```

-- 11. Listar todos los alquileres y sus respectivos clientes.

```
SELECT a.id_alquiler, c.nombre1, c.apellido1, a.fecha_salida, a.fecha_llegada  
FROM Alquileres a  
JOIN Clientes c ON a.id_cliente = c.id_cliente;
```

-- 12. Clientes que no han realizado alquileres.

```
SELECT * FROM Clientes WHERE id_cliente NOT IN (SELECT id_cliente FROM  
Alquileres);
```

-- 13. Vehículos con motor V8.

```
SELECT * FROM Vehiculos WHERE motor = 'V8';
```

-- 14. Total de ingresos por alquileres:

```
SELECT SUM(valor_pagado) AS ingresos_totales FROM Alquileres;
```

-- 15. Listar vehiculos de un tipo.

```
SELECT * from Vehiculos v inner join Tipo_vehiculo t on v.id_tipoV = t.id_tipoV where  
v.id_tipoV = 3;
```

-- 16. Vehículos más alquilados:

```
SELECT id_vehiculo, COUNT(*) AS veces_alquilado  
FROM Alquileres  
GROUP BY id_vehiculo  
ORDER BY veces_alquilado DESC;
```

-- 17. Listar todos los descuentos que superen el 15%.

```
SELECT * FROM Descuentos WHERE porcentaje_descuento > 15;
```

-- 18. Listar clientes de una ciudad específica.



```
SELECT * FROM Clientes WHERE ciudad = 'Bogotá';
```

-- 19. Cantidad de vehículos disponibles por modelo.

```
SELECT referencia, COUNT(*) AS cantidad FROM Vehiculos GROUP BY referencia;
```

-- 20. Alquileres ordenados por fecha de inicio:

```
SELECT * FROM Alquileres ORDER BY fecha_salida DESC;
```

-- 21. Clientes cuyo nombre comienza con 'A':

```
SELECT * FROM Clientes WHERE nombre1 LIKE 'A%';
```

-- 22. Alquileres con costo total mayor a \$500:

```
SELECT * FROM Alquileres WHERE valor_pagado > 500;
```

-- 23. Descuentos vigentes para un tipo de vehículo específico:

```
SELECT * FROM Descuentos WHERE id_tipoV = 2 AND '2024-11-18' BETWEEN  
fecha_inicio AND fecha_fin;
```

# Funciones

Es un objeto que se crea con la sentencia CREATE FUNCTION y se invoca con la sentencia SELECT o dentro de una expresión. Una función puede tener cero o muchos parámetros de entrada y siempre devuelve un valor, asociado al nombre de la función.

- 1 funcion

```
delimiter //
create function semanas_dias(fecha_inicio date, fecha_cierre date)
returns varchar(150) deterministic
begin
    declare cant_dias_semanas varchar(150);
    declare cant_dias int;
    declare cant_semanas int;
    declare diferencia_dias int;

    set diferencia_dias=TIMESTAMPDIFF(day, fecha_inicio, fecha_cierre);

    set cant_semanas = floor(diferencia_dias/7);

    set cant_dias= diferencia_dias%7;

    set cant_dias_semanas=concat('cantidad de semanas: ', cant_semanas, ' cantidad de
dias: ', cant_dias);

    return cant_dias_semanas;

end // delimiter ;
```

```
select id_alquiler, semanas_dias(fecha_salida, fecha_llegada) from alquileres;
```

- 2 funcion

Calcular la cantidad de días de retraso en la entrega de un alquiler

```
DELIMITER //

CREATE FUNCTION total_ingresos_por_vehiculo(vehiculo_id INT)
RETURNS INT deterministic
BEGIN
    DECLARE total_ingresos INT;

    SELECT SUM(valor_pagado)
```

```

        INTO total_ingresos
        FROM Alquileres
        WHERE id_vehiculo = vehiculo_id;

        RETURN IFNULL(total_ingresos, 0);
    END //

DELIMITER ;

SELECT total_ingresos_por_vehiculo(1) AS ingresos;

-- Calcular la cantidad de días de retraso en la entrega de un alquiler

DELIMITER //

```

- 3 funcion  
Calcular el total de ingresos por un vehículo específico

```

DELIMITER //

CREATE FUNCTION total_ingresos_por_vehiculo(vehiculo_id INT)
RETURNS INT
BEGIN
    DECLARE total_ingresos INT;

    SELECT SUM(valor_pagado)
    INTO total_ingresos
    FROM Alquileres
    WHERE id_vehiculo = vehiculo_id;

    RETURN IFNULL(total_ingresos, 0);
END //

DELIMITER ;

SELECT total_ingresos_por_vehiculo(1) AS ingresos;

-- 4 funcion
-- Obtener el nombre completo de un cliente basado en su ID

DELIMITER //

CREATE FUNCTION nombre_completo_cliente(id_cliente INT)

```

```

RETURNS VARCHAR(200) deterministic
BEGIN
    DECLARE nombre_completo VARCHAR(200);

    SELECT CONCAT(nombre1, ' ', nombre2, ' ', apellido1, ' ', apellido2)
    INTO nombre_completo
    FROM Clientes
    WHERE id_cliente = id_cliente;

    RETURN nombre_completo;
END //

DELIMITER ;

SELECT nombre_completo_cliente(1) AS nombre;

```

- 5 funcion

```

DELIMITER //

CREATE FUNCTION contar_vehiculos_por_tipo(tipo_vehiculo_id INT)
RETURNS INT deterministic
BEGIN
    DECLARE cantidad INT;

    SELECT COUNT(*)
    INTO cantidad
    FROM Vehiculos
    WHERE id_tipoV = tipo_vehiculo_id;

    RETURN IFNULL(cantidad, 0);
END //

DELIMITER ;

SELECT contar_vehiculos_por_tipo(2) AS vehiculos_disponibles;

```

# Procedimientos

Se han desarrollado una serie de procedimientos destinados a facilitar al usuario la inserción de datos en diversas tablas, asegurando así la integridad y consistencia de la base de datos mediante el mantenimiento adecuado de las relaciones entre ellas.

## 1. **Procedimiento:** InsertarSucursal

### a. **Descripción:**

Este procedimiento se creó con el fin de que se pueda mantener un buen registro en la tabla de entidades asegurándose de cumplir con el tipo de dato.

### b. **Sintaxis:**

```
DELIMITER //
CREATE PROCEDURE InsertarSucursal (
    IN id_sucursal INT,
    IN ubicacion VARCHAR(50),
    IN direccion VARCHAR(50),
    IN ciudad VARCHAR(50),
    IN telefono_fijo VARCHAR(50),
    IN celular VARCHAR(50),
    IN correo_electronico VARCHAR(255)
)
BEGIN
    INSERT INTO Sucursales (id_sucursal, ubicacion, direccion, ciudad,
telefono_fijo, celular, correo_electronico)
VALUES (id_sucursal, ubicacion, direccion, ciudad, telefono_fijo, celular,
correo_electronico);
END //
DELIMITER ;
```

### c. **Parámetros:**

IN id\_sucursal INT, IN ubicacion VARCHAR(50), IN direccion VARCHAR(50), IN ciudad VARCHAR(50), IN telefono\_fijo VARCHAR(50), IN celular VARCHAR(50), IN correo\_electronico VARCHAR(255)

**d. Retorno:**

El procedimiento va a retornar un insert hacia la tabla de entidad

**e. Ejemplo de implementación:**

call InsertarSucursal(6, 'Norte', 'Medellín', 'Carrera 15 #85-24', '6047654321', '3159876543', 'norte@sucursal.com');

## 2. Procedimiento: InsertarEmpleado

**a. Descripción:**

Este procedimiento se creó con el fin de que se pueda mantener un buen registro en la tabla de entidades asegurándose de cumplir con el tipo de dato.

**b. Sintaxis:**

```
DELIMITER //
CREATE PROCEDURE InsertarEmpleado (
    IN id_employado INT,
    IN cedula VARCHAR(50),
    IN nombre1 VARCHAR(60),
    IN nombre2 VARCHAR(60),
    IN apellido1 VARCHAR(60),
    IN apellido2 VARCHAR(60),
    IN direccion VARCHAR(50),
    IN ciudad VARCHAR(50),
    IN celular VARCHAR(50),
    IN correo_electronico VARCHAR(50),
    IN id_sucursal INT
)
BEGIN
    INSERT INTO Empleados (id_employado, cedula, nombre1, nombre2,
    apellido1, apellido2, direccion, ciudad, celular, correo_electronico, id_sucursal)
    VALUES (id_employado, cedula, nombre1, nombre2, apellido1, apellido2,
    direccion, ciudad, celular, correo_electronico, id_sucursal);
END //
DELIMITER ;
```

**c. Parámetros:**

```
IN id_empleado INT, IN cedula VARCHAR(50), IN nombre1 VARCHAR(60), IN
nombre2 VARCHAR(60), IN apellido1 VARCHAR(60), IN apellido2
VARCHAR(60), IN direccion VARCHAR(50), IN ciudad VARCHAR(50), IN celular
VARCHAR(50), IN correo_electronico VARCHAR(50), IN id_sucursal INT
```

d. **Retorno:** El procedimiento va a retornar un insert hacia la tabla de entidad

e. **Ejemplo de implementación:**

```
call InsertarEmpleado (101, '1029876543', 'Zully', 'Fernanda', 'Ortíz', 'Avendaño',
'Carrera 50 #10-20', 'Medellín', '3147896541', 'zullyorti@gmail.com', 2);
```

### 3. **Procedimiento:** InsertarVehiculo

a. **Descripción:**

Este procedimiento se creó con el fin de que se pueda mantener un buen registro en la tabla de entidades asegurándose de cumplir con el tipo de dato.

b. **Sintaxis:**

```
DELIMITER //
CREATE PROCEDURE InsertarVehiculo (
    IN id_vehiculo INT,
    IN placa VARCHAR(50),
    IN referencia INT,
    IN modelo VARCHAR(50),
    IN puertas INT,
    IN capacidad INT,
    IN sunroof VARCHAR(30),
    IN motor VARCHAR(50),
    IN color VARCHAR(40),
    IN id_tipoV INT
)
BEGIN
    INSERT INTO Vehiculos (id_vehiculo, placa, referencia, modelo, puertas,
    capacidad, sunroof, motor, color, id_tipoV)
    VALUES (id_vehiculo, placa, referencia, modelo, puertas, capacidad, sunroof,
    motor, color, id_tipoV);
END //
DELIMITER ;
```

c. **Parámetros:**

IN id\_vehiculo INT, IN placa VARCHAR(50), IN referencia INT, IN modelo VARCHAR(50), IN puertas INT, IN capacidad INT, IN sunroof VARCHAR(30), IN motor VARCHAR(50), IN color VARCHAR(40), IN id\_tipoV INT.

d. **Retorno:** El procedimiento va a retornar un insert hacia la tabla de entidad

e. **Ejemplo de implementación:**

call InsertarVehiculo (101, 'DEF456', 2022, 'Toyota Corolla', 4, 5, 'No', '1.8L', 'Blanco', 2);

#### 4. **Procedimiento:** InsertarTipoVehiculo

a. **Descripción:**

Este procedimiento se creó con el fin de que se pueda mantener un buen registro en la tabla de entidades asegurándose de cumplir con el tipo de dato.

b. **Sintaxis:**

```
DELIMITER //
CREATE PROCEDURE InsertarTipoVehiculo(
    IN id_tipoV INT,
    IN valor_alquiler_semana INT,
    IN valor_alquiler_dia INT,
    IN tipo VARCHAR(55)
)
BEGIN
    INSERT INTO tipo_vehiculo (id_tipoV, valor_alquiler_semana,
    valor_alquiler_dia, tipo )
    VALUES (id_tipoV, valor_alquiler_semana, valor_alquiler_dia, tipo);
END //
DELIMITER ;
```

c. **Parámetros:**

IN id\_tipoV INT, IN valor\_alquiler\_semana INT, IN valor\_alquiler\_dia INT, IN tipo VARCHAR(55)

d. **Retorno:** El procedimiento va a retornar un insert hacia la tabla de entidad



**e. Ejemplo de implementación:**

```
call InsertarTipoVehiculo(11, 300000, 18000,'Furgoneta');
```

**5. Procedimiento: InsertarClientes**

**a. Descripción:**

Este procedimiento se creó con el fin de que se pueda mantener un buen registro en la tabla de entidades asegurándose de cumplir con el tipo de dato.

**b. Sintaxis:**

```
CREATE PROCEDURE InsertarClientes (  
    IN id_cliente INT,  
    IN cedula VARCHAR(50),  
    IN nombre1 VARCHAR(60),  
    IN nombre2 VARCHAR(60),  
    IN apellido1 VARCHAR(60),  
    IN apellido2 VARCHAR(60),  
    IN direccion VARCHAR(50),  
    IN ciudad VARCHAR(50),  
    IN celular VARCHAR(20),  
    IN correo_electronico VARCHAR(50)  
)  
BEGIN  
    INSERT INTO Clientes (id_cliente,cedula, nombre1, nombre2, apellido1,  
    apellido2, direccion, ciudad, celular, correo_electronico)  
    VALUES (id_cliente,cedula, nombre1, nombre2, apellido1, apellido2, direccion,  
    ciudad, celular, correo_electronico);  
END //  
DELIMITER ;
```

**c. Parámetros:**

```
IN id_cliente INT, IN cedula VARCHAR(50), IN nombre1 VARCHAR(60), IN  
nombre2 VARCHAR(60), IN apellido1 VARCHAR(60), IN apellido2
```

`VARCHAR(60), IN direccion VARCHAR(50), IN ciudad VARCHAR(50), IN celular VARCHAR(20), IN correo_electronico VARCHAR(50)`

d. **Retorno:** El procedimiento va a retornar un insert hacia la tabla de entidad

e. **Ejemplo de implementación:**

`call InsertarClientes (101, '10987654321', 'Yessica', 'Andrea', 'Machuca', 'Perez', 'Calle 10', 'Bogotá', 3001234033, 'yessica.Machuca@cliente.com');`

## 6. **Procedimiento:** ActualizarCliente

a. **Descripción:**

Este procedimiento se creó con el fin de que se pueda mantener un buen registro en la tabla de entidades asegurándose de cumplir con el tipo de dato.

b. **Sintaxis:**

```
DELIMITER //
CREATE PROCEDURE ActualizarCliente (
    IN id_cliente1 INT,
    IN celular1 VARCHAR(20),
    IN direccion1 VARCHAR(50),
    IN correo_electronico1 VARCHAR(50)
)
BEGIN
    UPDATE Clientes
    SET celular = celular1,
        direccion = direccion1,
        correo_electronico = correo_electronico1
    WHERE id_cliente = id_cliente1;
END //
DELIMITER ;
```

c. **Parámetros:**

`IN id_cliente1 INT, IN celular1 VARCHAR(20), IN direccion1 VARCHAR(50), IN correo_electronico1 VARCHAR(50)`

d. **Retorno:** El procedimiento va a retornar un insert hacia la tabla de entidad

e. **Ejemplo de implementación:**

`call ActualizarCliente (1, '10987654345', 'Tibu', freiler.ortega@cliente.com);`

## 7. **Procedimiento:** EliminarDescuento

### a. **Descripción:**

Este procedimiento se creó con el fin de que se pueda mantener un buen registro en la tabla de entidades asegurándose de cumplir con el tipo de dato.

### b. **Sintaxis:**

```
DELIMITER //  
CREATE PROCEDURE EliminarDescuento (  
    IN id_descuento1 INT  
)  
BEGIN  
    DELETE FROM Descuentos  
    WHERE id_descuento = id_descuento1;  
END //  
DELIMITER ;
```

### c. **Parámetros:**

`IN id_descuento1 INT`

**d. Retorno:** El procedimiento va a retornar un insert hacia la tabla de entidad

### e. **Ejemplo de implementación:**

`call EliminarDescuento (1);`

## 8. Procedimiento: ListarSucursal

### a. Descripción:

Este procedimiento se creó con el fin de que se pueda mantener un buen registro en la tabla de entidades asegurándose de cumplir con el tipo de dato.

### b. Sintaxis:

```
DELIMITER //
CREATE PROCEDURE ListarSucursal (
    IN id_sucursal INT
)
BEGIN
    SELECT e.nombre1, e.apellido1, e.celular, e.correo_electronico
    FROM Empleados e
    WHERE e.id_sucursal = id_sucursal;
END //
DELIMITER ;
```

### c. Parámetros:

IN id\_sucursal INT

### d. Retorno:

El procedimiento va a retornar un insert hacia la tabla de entidad

### e. Ejemplo de implementación:

call ListarSucursal(2);

## 9. Procedimiento: descuento

### a. Descripción:

Este procedimiento se creó con el fin de que se pueda mantener un buen registro en la tabla de entidades asegurándose de cumplir con el tipo de dato.

### b. Sintaxis:

```
DELIMITER //
CREATE PROCEDURE descuento(
    IN id_descuento INT,
    IN fecha_inicio DATE,
    IN fecha_fin DATE,
    IN porcentaje_descuento INT,
    IN id_tipoV INT
)
BEGIN
    INSERT INTO Descuentos(id_descuento, fecha_inicio, fecha_fin,
    porcentaje_descuento,id_tipoV)
    VALUES (id_descuento, fecha_inicio, fecha_fin,
    porcentaje_descuento,id_tipoV);
END
// DELIMITER ;
```

### c. Parámetros:

IN id\_descuento INT, IN fecha\_inicio DATE, IN fecha\_fin DATE, IN porcentaje\_descuento INT, IN id\_tipoV INT.

d. **Retorno:** El procedimiento va a retornar un insert hacia la tabla de entidad

e. **Ejemplo de implementación:**

```
call ListarSucursal(2);
```

## 10. **Procedimiento:** AplicarDescuento

a. **Descripción:** Este procedimiento se creó con el fin de que se pueda mantener un buen registro en la tabla de entidades asegurándose de cumplir con el tipo de dato.

b. **Sintaxis:**

```
DELIMITER //  
CREATE PROCEDURE descuento(  
    IN id_descuento INT,  
    IN fecha_inicio DATE,  
    IN fecha_fin DATE,  
    IN porcentaje_descuento INT,  
    IN id_tipoV INT)  
BEGIN  
    INSERT INTO Descuentos(id_descuento, fecha_inicio, fecha_fin,  
porcentaje_descuento,id_tipoV)  
    VALUES (id_descuento, fecha_inicio, fecha_fin,  
porcentaje_descuento,id_tipoV);  
END  
// DELIMITER ;
```

c. **Parámetros:**

IN id\_descuento INT, IN fecha\_inicio DATE, IN fecha\_fin DATE, IN porcentaje\_descuento INT, IN id\_tipoV INT.

d. **Retorno:** El procedimiento va a retornar un insert hacia la tabla de entidad

e. **Ejemplo de implementación:**

CALL AplicarDescuento(1, 10);