Instituto Politécnico Nacional

Centro de Estudios Científicos y Tecnológicos 9

"Juan de Dios Bátiz"

PRÁCTICA INTEGRADORA

ALUMNOS

Andres Marin Carlos

Recillas Palacios Raúl

GRUPO

4IM8

UNIDAD DE APRENDIZAJE

Base de Datos

PROFESOR/A

Rosa Iliana Fuentes Cruz

FECHA DE ENTREGA

13 de junio de 2024

ÍNDICE

INTRODUCCIÓN

En el siguiente documento se muestra el proceso de diseño para la base de datos de un autolavado desde la planeación hasta la codificación de los scripts de esta.

Para poder realizar esta base de datos primero analizamos los requerimientos, después propusimos un modelo para la base de datos y al final realizamos los procedimientos y triggers para que pueda ser una base de datos funcional.

Nuestro objetivo principal fue realizar una base de datos lo más simplificada posible para facilitar el manejo de datos, pero sin tener que perder funcionalidad.

Uno de los grandes retos que nos encontramos al momento de diseñar la base de datos, fue optimizarla y simplificarla para evitar tener muchas tablas que puedan ser redundantes y nos compliquen el manejo de datos.

ANÁLISIS DE LA SITUACIÓN

DEFINICIÓN DEL PROBLEMA

El cliente solicita facilitar el control y seguimiento en la operación diaria del negocio del auto lavado, ventas, catálogo de productos (paquetes ofrecidos), promociones, clientes, administración de empleados, sueldos de empleados, membresías e información de últimos servicios realizados al automóvil, así como los respectivos reportes para cada sección.

Realizar el diseño de una base de datos que considere lo anterior y facilite la gestión del auto lavado considerando también los siguientes puntos:

- Manejo de sucursales (mínimo 5)
- Control de empleados (datos personales, sueldos, horarios, etc.)
- Manejo y control de Promociones (Cliente frecuente, seguro de lluvia, 2 x 1, Cortesías, etc.) Las promociones pueden existir un una o más sucursales, cada sucursal debe tener como mínimo 2 promociones.
- Manejo de Membresías (ejemplo: Por tiempo o Número de visitas, deben existir mínimo 3 tipos de membresías). Si el cliente cuenta con una puede acceder a cualquier sucursal y promociones.
- Registro de datos del cliente y comentarios.
- Gestión de productos (paquetes ofrecidos, tipo de servicios)
- Manejo de tipos de pago (efectivo, tarjeta o promoción)
- Para clientes que cambian de auto, permitir integrar las ventas los datos de las placas asociadas al cliente para que no pierda las ventajas de ser cliente frecuente.
- Permitir llevar el registro de los comentarios de los clientes, de manera que la próxima vez que el cliente se presente, sea posible proporcionar una atención más puntual en la falla incurrida.

ANÁLISIS DE REQUERIMIENTOS

Para poder llevar a cabo el diseño de la base de datos debemos resaltar los siguientes puntos importantes.

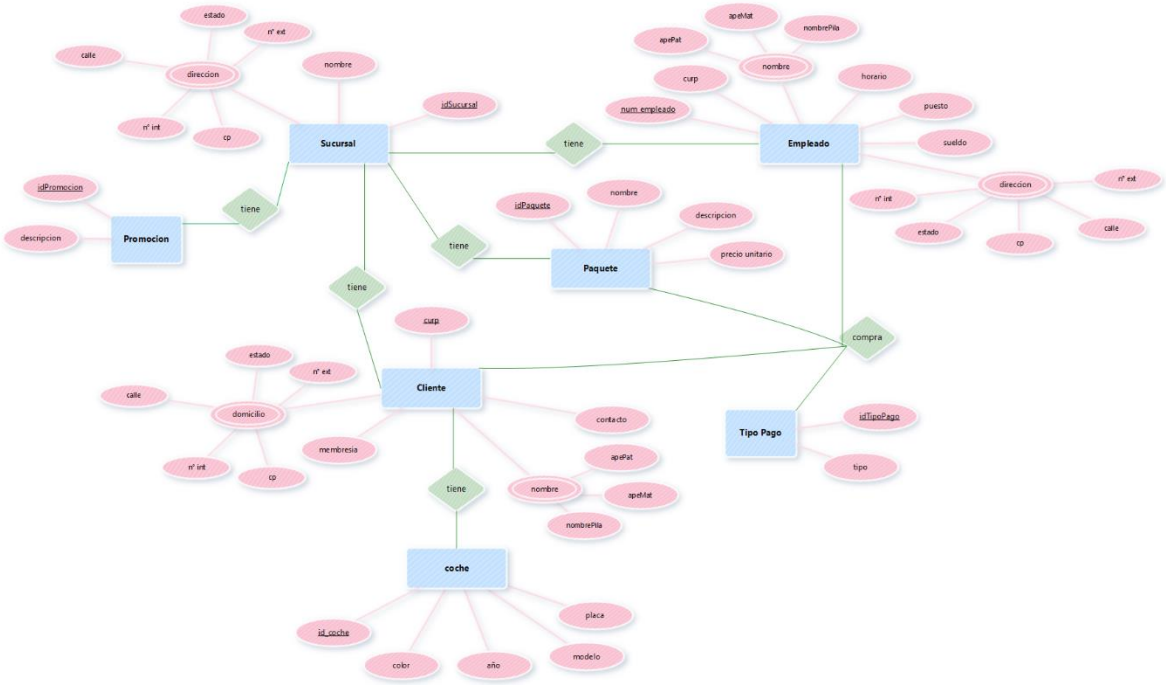
- La base de datos debe de cubrir las necesidades del cliente
- La base de datos debe de tener la capacidad de recabar la información de una forma eficiente y eficaz
- Debe de ser fácil el manejo de la información
- No debe de haber información duplicada

Tomando en cuenta los puntos anteriores, nuestra base de datos va a tener 8 entidades principales, estas son las siguientes:

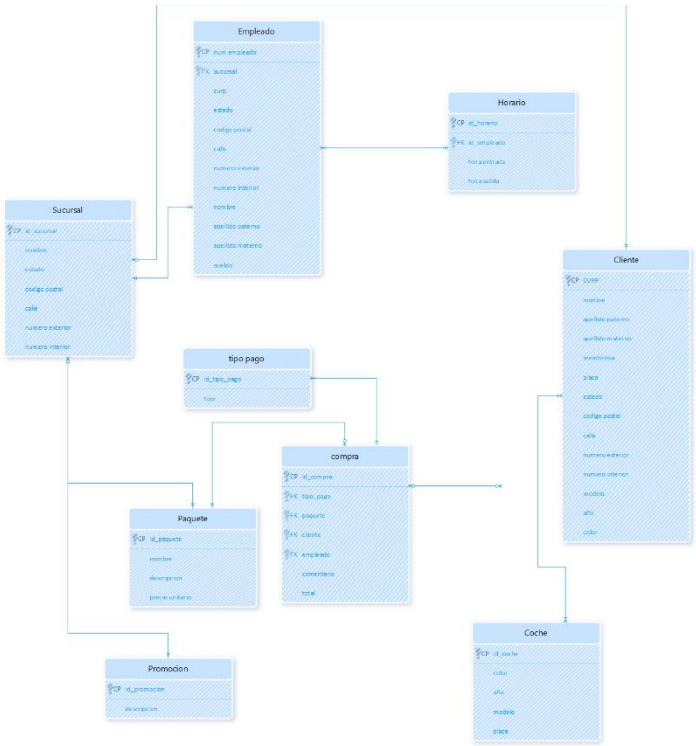
- Sucursal
- Empleado
- Cliente
- Paquete
- Coche
- Tipo de pago
- Promocion

DISEÑO DE LA BASE DE DATOS

MODELO ENTIDAD RELACIÓN



MODELO RELACIONAL



NORMALIZACIÓN DE BASE DE DATOS

Sucursal

id_sucursal	id_direccion	nombre

Estado

id_estado	estado

Cliente

curp	membresia	domicilio	nombre	apellido_paterno	apellido_materno

Direccion

id_direccion	estado	codigo_postal	calle	numero_exterior	numero_interior

Coche

placa	cliente	modelo	año	color

Promocion

id_promocion	descripcion	condicion

Rel_Promo_Sucur

id_rel	id_sucursal	id_promocion

Cronograma

id_cronograma	id_empleado	id_horario

Compra

id_compra	id_paquete	id_ticket

Ticket

id_ticket	cliente	operador	coche	sucursal	tipo_pago	descuento	comentario	fecha	subtotal	total

Membresia

id_membresia	membresia	condicion

Contacto

id_contacto	tipo	contacto

Puesto

id_puesto	puesto	salario

Tipo_Contacto

id_cat	tipo

Empleado

num_empleado	id_sucursal	id_puesto	domicilio	curp	nombre	apellido_paterno	apellido_materno

Horario

id_horario	id_cronograma	hora_entrada	hora_salida

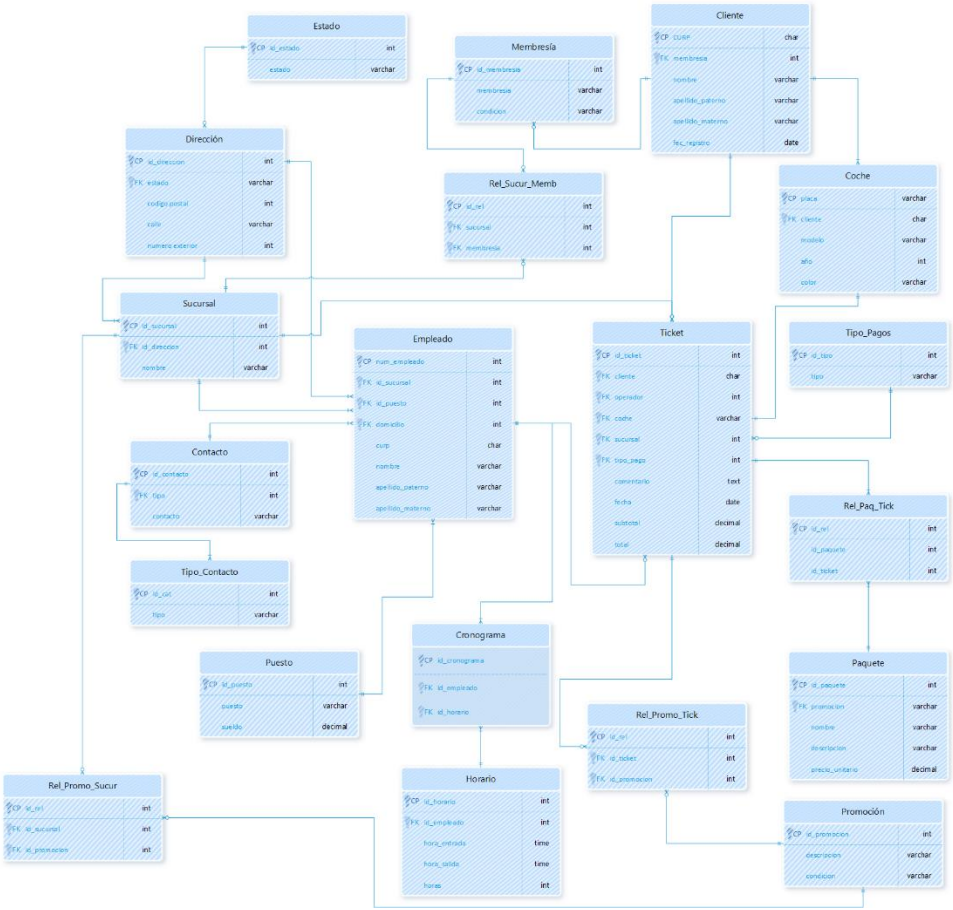
Paquete

id_paquete	nombre	descripcion	precio_unitario

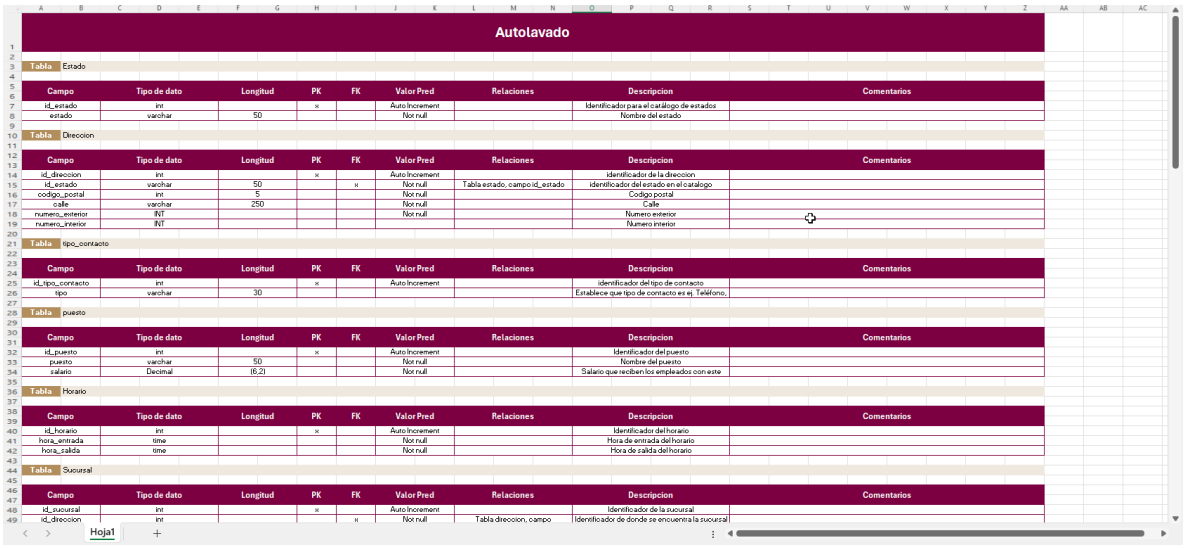
Tipo_Pago

id_tipo	tipo

MODELO RELACIONAL NORMALIZADO



DICCIONARIO DE DATOS



Autolavado

Tabla Estado

Campo	Tipo de dato	Longitud	PK	FK	Valor Pred	Relaciones	Descripcion	Comentarios
id_estado	int		x		Auto Increment		Identificador para el catálogo de estados	
estado	varchar	50			Not null		Nombre del estado	

Tabla Direccion

Campo	Tipo de dato	Longitud	PK	FK	Valor Pred	Relaciones	Descripcion	Comentarios
id_direccion	int		x		Auto Increment		identificador de la direccion	
id_estado	varchar	50		x	Not null	Tabla estado, campo id_estado	identificador del estado en el catalogo	
codigo_postal	int	5			Not null		Codigo postal	
calle	varchar	250			Not null		Calle	
numero_exterior	INT				Not null		Numero exterior	
numero_interior	INT						Numero interior	

Tabla tipo_contacto

Campo	Tipo de dato	Longitud	PK	FK	Valor Pred	Relaciones	Descripcion	Comentarios
id_tipo_contacto	int		x		Auto Increment		identificador del tipo de contacto	
tipo	varchar	30					Establece que tipo de contacto es ej. Teléfono,	

Tabla puesto

Campo	Tipo de dato	Longitud	PK	FK	Valor Pred	Relaciones	Descripcion	Comentarios
id_puesto	int		x		Auto Increment		Identificador del puesto	
puesto	varchar	50			Not null		Nombre del puesto	
salario	Decimal	(6,2)			Not null		Salario que reciben los empleados con este	

Tabla Horario

Campo	Tipo de dato	Longitud	PK	FK	Valor Pred	Relaciones	Descripcion	Comentarios
id_horario	int		x		Auto Increment		Identificador del horario	
hora_entrada	time				Not null		Hora de entrada del horario	
hora_salida	time				Not null		Hora de salida del horario	

Tabla Sucursal

Campo	Tipo de dato	Longitud	PK	FK	Valor Pred	Relaciones	Descripcion	Comentarios
id_sucursal	int		x		Auto Increment		Identificador de la sucursal	
id_direccion	int			x	Not null	Tabla direccion, campo	Identificador de donde se encuentra la sucursal	

Hoja1

Tabla Empleado

Campo	Tipo de dato	Longitud	PK	FK	Valor Pred	Relaciones	Descripcion	Comentarios
numero_empleado	int		x		Auto Increment		Identificador del empleado	
id_sucursal	int			x	Not null	Tabla sucursal, campo id_sucursal	Identificador de la sucursal donde trabaja	
id_puesto	int			x	Not null	Tabla puesto, campo id_puesto	identificador del puesto que desempeña	
nombre	varchar	100			Not null		Nombre del empleado	
apellido_paterno	varchar	50			Not null		Apellido paterno del empleado	
apellido_materno	varchar	50			Not null		Apellido materno del empleado	

Tabla Cronograma

Campo	Tipo de dato	Longitud	PK	FK	Valor Pred	Relaciones	Descripcion	Comentarios
id_cronograma	int		x		Auto Increment		Identificador del empleado	
numero_empleado	int			x	Not null	Tabla empleado, campo numero_empleado	Numero del empleado con este cronograma	
id_horario	int			x	Not null	Tabla puesto, campo id_puesto	Identificador del horario correspondiente a este cronograma	Con cronograma nos referimos a uno de los "turnos" del empleado, los empleados pueden tener multiples horarios asignados. Esta es la tabla de relacion entre horarios y empleados

Tabla Membresia

Campo	Tipo de dato	Longitud	PK	FK	Valor Pred	Relaciones	Descripcion	Comentarios
id_membresia	int		x		Auto Increment		Identificador de la membresia	
membresia	varchar	50			Not null		nombre de la membresia	
condicion	varchar	250			Not null		condicion para aplicar la membresia	

Tabla Promocion

Campo	Tipo de dato	Longitud	PK	FK	Valor Pred	Relaciones	Descripcion	Comentarios
id_promocion	int		x		Auto Increment		Identificador de la promocion	
descripcion	varchar	50			Not null		nombre de la promocion	
condicion	varchar	250			Not null		condicion para aplicar la promocion	

Tabla Cliente

Campo	Tipo de dato	Longitud	PK	FK	Valor Pred	Relaciones	Descripcion	Comentarios
curp	char	18	x		Auto Increment		Identificador del cliente	
id_membresia	int			x	Not null	Tabla membresia, campo id_membresia	Identificador de la membresia con la que cuenta el cliente	
id_direccion	int			x	Not null	Tabla direccion, id_direccion	Identificador del domicilio del cliente	
nombre	varchar	100			Not null		Nombre del cliente	
apellido_paterno	varchar	50			Not null		Apellido paterno del cliente	
apellido_materno	varchar	50			Not null		Apellido materno del cliente	
fecha_registro	date				Not null		Fecha en la que el cliente fue registrado	

Tabla Contacto

Hoja1

DISEÑO LÓGICO

CREACIÓN DE LA BASE DE DATOS Y TABLAS EN MYSQL

El script utilizado para la creación de la base de datos es el siguiente:

```
DROP DATABASE IF EXISTS Autolavado;

CREATE DATABASE Autolavado;

USE Autolavado;


CREATE TABLE estado (

  id_estado INT PRIMARY KEY AUTO_INCREMENT,
```

```
  estado VARCHAR(50) NOT NULL
);

CREATE TABLE direccion (
  id_direccion INT PRIMARY KEY AUTO_INCREMENT,
  id_estado INT NOT NULL,
  codigo_postal INT(5) NOT NULL,
  calle VARCHAR(250) NOT NULL,
  numero_exterior INT NOT NULL,
  numero_interior INT,
  FOREIGN KEY (id_estado) REFERENCES estado(id_estado)
);

CREATE TABLE tipo_contacto (
  id_tipo_contacto INT PRIMARY KEY AUTO_INCREMENT,
  tipo VARCHAR(30)
);

CREATE TABLE puesto (
  id_puesto INT PRIMARY KEY AUTO_INCREMENT,
  puesto VARCHAR(50) NOT NULL,
  salario DECIMAL (6,2) NOT NULL
);

CREATE TABLE horario (
  id_horario INT PRIMARY KEY AUTO_INCREMENT,
  hora_entrada TIME NOT NULL,
  hora_salida TIME NOT NULL
);

CREATE TABLE sucursal (
  id_sucursal INT PRIMARY KEY AUTO_INCREMENT,
  id_direccion INT NOT NULL,
```

```
  nombre VARCHAR(255) NOT NULL,
  FOREIGN KEY (id_direccion) REFERENCES direccion(id_direccion)
);

CREATE TABLE empleado (
  numero_empleado INT PRIMARY KEY AUTO_INCREMENT,
  id_sucursal INT NOT NULL,
  id_puesto INT NOT NULL,
  nombre VARCHAR(100) NOT NULL,
  apellido_paterno VARCHAR(50) NOT NULL,
  apellido_materno VARCHAR(50) NOT NULL,
  FOREIGN KEY (id_sucursal) REFERENCES sucursal(id_sucursal),
  FOREIGN KEY (id_puesto) REFERENCES puesto(id_puesto)
);

CREATE TABLE cronograma (
  id_cronograma INT PRIMARY KEY AUTO_INCREMENT,
  numero_empleado INT NOT NULL,
  id_horario INT NOT NULL,
  FOREIGN KEY (numero_empleado) REFERENCES empleado(numero_empleado),
  FOREIGN KEY (id_horario) REFERENCES horario(id_horario)
);

CREATE TABLE membresia (
        id_membresia INT PRIMARY KEY AUTO_INCREMENT,
        membresia VARCHAR(50) NOT NULL,
        condicion VARCHAR(250) NOT NULL
);

CREATE TABLE promocion (
        id_promocion INT PRIMARY KEY AUTO_INCREMENT,
        descripcion VARCHAR(50) NOT NULL,
        condicion VARCHAR(250) NOT NULL
```

```
);

CREATE TABLE cliente (
	curp CHAR(18) PRIMARY KEY,
	id_membresia INT NOT NULL,
	id_direccion INT NOT NULL,
	nombre VARCHAR(100) NOT NULL,
	apellido_paterno VARCHAR(50) NOT NULL,
	apellido_materno VARCHAR(50) NOT NULL,
	fecha_registro DATE,
	FOREIGN KEY (id_membresia) REFERENCES membresia(id_membresia),
	FOREIGN KEY (id_direccion) REFERENCES direccion(id_direccion)
);

CREATE TABLE contacto (
  id_contacto INT PRIMARY KEY AUTO_INCREMENT,
  curp CHAR(18) NOT NULL,
  id_tipo_contacto INT NOT NULL,
  contacto VARCHAR(76),
  FOREIGN KEY (curp) REFERENCES cliente(curp),
  FOREIGN KEY (id_tipo_contacto) REFERENCES tipo_contacto(id_tipo_contacto)
);

CREATE TABLE tipos_pago (
	id_tipos_pago INT PRIMARY KEY AUTO_INCREMENT,
	tipo VARCHAR(25) NOT NULL
);

CREATE TABLE paquete (
	id_paquete INT PRIMARY KEY AUTO_INCREMENT,
	descripcion VARCHAR(250),
	precio DECIMAL(6,2)
);
```

```
CREATE TABLE coche (
        placa CHAR(8) PRIMARY KEY,
        curp CHAR(18) NOT NULL,
        modelo VARCHAR(50) NOT NULL,
        año INT NOT NULL,
        color VARCHAR(25) NOT NULL,
        FOREIGN KEY (curp) REFERENCES cliente(curp)
);

CREATE TABLE ticket (
        id_ticket INT PRIMARY KEY AUTO_INCREMENT,
        cliente CHAR(18) NOT NULL,
        operador INT NOT NULL,
        coche CHAR(8) NOT NULL,
        sucursal INT NOT NULL,
        tipo_pago INT NOT NULL,
        paquete INT NOT NULL,
        promocion INT,
        comentario VARCHAR(250),
        subtotal DECIMAL(8,2),
        total DECIMAL(8,2),
        FOREIGN KEY (cliente) REFERENCES cliente(curp),
        FOREIGN KEY (operador) REFERENCES empleado(numero_empleado),
  FOREIGN KEY (coche) REFERENCES coche(placa),
        FOREIGN KEY (sucursal) REFERENCES sucursal(id_sucursal),
        FOREIGN KEY (tipo_pago) REFERENCES tipos_pago(id_tipos_pago),
        FOREIGN KEY (paquete) REFERENCES paquete(id_paquete),
        FOREIGN KEY (promocion) REFERENCES promocion(id_promocion)
);

CREATE TABLE compra (
        id_compra INT PRIMARY KEY AUTO_INCREMENT,
```

```
        id_ticket INT NOT NULL,
        id_paquete INT NOT NULL,
        cantidad INT,
        precio DECIMAL(6,2),
        FOREIGN KEY (id_ticket) REFERENCES ticket(id_ticket),
        FOREIGN KEY (id_paquete) REFERENCES paquete(id_paquete)
);

CREATE TABLE promocion_sucursal (
        id_promocion_sucursal INT PRIMARY KEY AUTO_INCREMENT,
  id_promocion INT NOT NULL,
  id_sucursal INT NOT NULL,
  fecha_inicio DATE NOT NULL,
  fecha_fin DATE NOT NULL,
  FOREIGN KEY (id_promocion) REFERENCES promocion(id_promocion),
  FOREIGN KEY (id_sucursal) REFERENCES sucursal(id_sucursal)
);
```

CREACIÓN DE PROCEDIMIENTOS ALMACENADOS

```
USE Autolavado;

DELIMITER //

CREATE PROCEDURE ReporteDiarioVentas()
BEGIN
  SELECT paquete.descripcion AS 'Tipo de servicio', COUNT(*) AS 'Numero de ventas', SUM(ticket.total) AS Total
  FROM ticket
  INNER JOIN paquete ON ticket.paquete = paquete.id_paquete
  WHERE DATE(ticket.fecha) = CURDATE()
  GROUP BY paquete.descripcion;
END; //
```

```
CREATE PROCEDURE BuscarCliente(
        IN $nombre VARCHAR(50)
)
BEGIN
  SELECT cliente.nombre, cliente.apellido_paterno, cliente.apellido_materno, membresia.membresia,
  paquete.descripcion AS 'Tipo de servicio', ticket.fecha
  FROM cliente
  INNER JOIN membresia ON cliente.id_membresia = membresia.id_membresia
  INNER JOIN ticket ON cliente.curp = ticket.cliente
  INNER JOIN paquete ON ticket.paquete = paquete.id_paquete
  WHERE cliente.nombre LIKE CONCAT('%', $nombre, '%')
  AND ticket.fecha >= DATE_SUB(CURDATE(), INTERVAL 10 DAY);
END; //

CREATE PROCEDURE BuscarPorNumero(
        IN $numero INT
)
BEGIN
  SELECT empleado.nempleado, empleado.nombre, empleado.apellido_paterno, empleado.apellido_materno, sucursal.nombre AS Sucursal,
  SUM(ticket.total) AS Total, AVG(ticket.total) AS Promedio
  FROM empleado
  INNER JOIN sucursal ON empleado.id_sucursal = sucursal.id_sucursal
  INNER JOIN ticket ON empleado.nempleado = ticket.operador
  WHERE empleado.nempleado = $numero
  AND ticket.fecha >= DATE_SUB(CURDATE(), INTERVAL 30 DAY)
  GROUP BY empleado.nempleado;
END; //

CREATE PROCEDURE OperadorDelMes()
BEGIN
```

```
  SELECT empleado.nempleado, empleado.nombre, sucursal.nombre AS Sucursal, SUM(ticket.total)
AS Total,
  CASE
    WHEN SUM(ticket.total) > 1000 THEN SUM(ticket.total) * 0.05
    ELSE 0
  END AS Bono,
  CASE
    WHEN SUM(ticket.total) > 1000 THEN 'Operador del mes'
    ELSE ''
  END AS Mensaje
  FROM empleado
  INNER JOIN sucursal ON empleado.id_sucursal = sucursal.id_sucursal
  INNER JOIN ticket ON empleado.nempleado = ticket.operador
  GROUP BY empleado.nempleado;
END; //

CREATE PROCEDURE ReporteServiciosCliente(
        IN $nombre VARCHAR(50)
)
BEGIN
  SELECT  cliente.nombre,  cliente.apellido_paterno,  cliente.apellido_materno,  paquete.descripcion
AS 'Tipo de servicio',
  COUNT(*) AS 'Numero de servicios'
  FROM ticket
  INNER JOIN cliente ON ticket.cliente = cliente.curp
  INNER JOIN paquete ON ticket.paquete = paquete.id_paquete
  INNER JOIN empleado ON ticket.operador = empleado.nempleado
  WHERE empleado.nombre LIKE CONCAT('%', $nombre, '%')
  GROUP BY cliente.curp, paquete.descripcion;
END; //

CREATE PROCEDURE AgregarSucursal(
  IN $nombre VARCHAR(50),
```

```
  IN $direccion VARCHAR(100),
  IN $telefono VARCHAR(15)
)
BEGIN
  DECLARE $existe INT DEFAULT 0;
  SELECT COUNT(*) INTO $existe
  FROM sucursal
  WHERE nombre = $nombre;
  IF $existe > 0 THEN
    SELECT 'La sucursal ya existe.';
  ELSE
    INSERT INTO sucursal (nombre, id_direccion, telefono)
    VALUES ($nombre, $direccion, $telefono);
    SELECT 'Sucursal agregada exitosamente.';
  END IF;
END; //

CREATE PROCEDURE AgregarEmpleado(
  IN $empleado INT,
  IN $sucursal INT,
  IN $puesto INT,
  IN $horario INT,
  IN $nombre VARCHAR(50),
  IN $apellido_paterno VARCHAR(50),
  IN $apellido_materno VARCHAR(50)
)
BEGIN
  DECLARE $existe INT DEFAULT 0;
  SELECT COUNT(*) INTO $existe
  FROM empleado
  WHERE numero_empleado = $empleado;
  IF $existe > 0 THEN
    SELECT 'El empleado ya existe.';
```

```
  ELSE
    START TRANSACTION;
    INSERT INTO empleado (numero_empleado, sucursal, puesto, nombre, apellido_paterno, apellido_materno)
    VALUES ($empleado, $sucursal, $puesto, $nombre, $apellido_paterno, $apellido_materno);
    INSERT INTO cronograma (numero_empleado, horario)
    VALUES ($empleado, $horario);
    COMMIT;
    SELECT 'Empleado agregado exitosamente.';
  END IF;
END; //

CREATE PROCEDURE AgregarCliente(
  IN $curp CHAR(18),
  IN $membresia INT,
  IN $direccion INT,
  IN $nombre VARCHAR(50),
  IN $apellido_paterno VARCHAR(50),
  IN $apellido_materno VARCHAR(50),
  IN $fecha DATE,
  IN $placa CHAR(8),
  IN $modelo VARCHAR(50),
  IN $ano INT,
  IN $color VARCHAR(25),
  IN $tipo_contacto INT,
  IN $contacto VARCHAR(76)
)
BEGIN
  DECLARE $cliente INT DEFAULT 0;
  DECLARE $coche INT DEFAULT 0;

  SELECT COUNT(*) INTO $cliente
  FROM cliente
```

```
  WHERE curp = $curp;

  SELECT COUNT(*) INTO $coche
  FROM coche
  WHERE placa = $placa;

  IF $cliente > 0 THEN
    SELECT 'El cliente ya existe.';
  ELSEIF $coche > 0 THEN
    SELECT 'El coche con esa placa ya existe.';
  ELSE
    START TRANSACTION;
    INSERT INTO cliente (curp, id_membresia, id_direccion, nombre, apellido_paterno, apellido_materno, fecha_registro)
    VALUES ($curp, $membresia, $direccion, $nombre, $apellido_paterno, $apellido_materno, $fecha);
    INSERT INTO coche (placa, curp, modelo, año, color)
    VALUES ($placa, $curp, $modelo, $ano, $color);
    INSERT INTO contacto (curp, id_tipo_contacto, contacto)
    VALUES ($curp, $tipo_contacto, $contacto);
    COMMIT;
    SELECT 'Cliente agregado exitosamente.';
  END IF;
END; //

CREATE PROCEDURE AgregarPromocionSucursal(
  IN $promocion INT,
  IN $sucursal INT,
  IN $fecha_inicio DATE,
  IN $fecha_fin DATE
)
BEGIN
  DECLARE $existe INT DEFAULT 0;
```

```
  SELECT COUNT(*) INTO $existe
  FROM promocion_sucursal
  WHERE id_promocion = $promocion AND id_sucursal = $sucursal;
  IF $existe > 0 THEN
    SELECT 'La promoción ya existe en la sucursal.';
  ELSE
    INSERT INTO promocion_sucursal (id_promocion, id_sucursal, fecha_inicio, fecha_fin)
    VALUES ($promocion, $sucursal, $fecha_inicio, $fecha_fin);
    SELECT 'Promoción agregada exitosamente a la sucursal.';
  END IF;
END; //

CREATE PROCEDURE AgregarHorario(
  IN $hora_entrada TIME,
  IN $hora_salida TIME
)
BEGIN
  INSERT INTO horario (hora_entrada, hora_salida)
  VALUES ($hora_entrada, $hora_salida);
  SELECT 'Horario agregado exitosamente.';
END; //

CREATE PROCEDURE AgregarContacto(
  IN $curp CHAR(18),
  IN $tipo_contacto INT,
  IN $contacto VARCHAR(76)
)
BEGIN
  INSERT INTO contacto (curp, id_tipo_contacto, contacto)
  VALUES ($curp, $tipo_contacto, $contacto);
  SELECT 'Contacto agregado exitosamente.';
END; //
```

```
CREATE PROCEDURE AgregarCoche(
  IN $placa CHAR(8),
  IN $curp CHAR(18),
  IN $modelo VARCHAR(50),
  IN $ano INT,
  IN $color VARCHAR(25)
)
BEGIN
  DECLARE $existe INT DEFAULT 0;

  SELECT COUNT(*) INTO $existe
  FROM coche
  WHERE placa = $placa;

  IF $existe > 0 THEN
    SELECT 'El coche con esa placa ya existe.';
  ELSE
    INSERT INTO coche (placa, curp, modelo, año, color)
    VALUES ($placa, $curp, $modelo, $ano, $color);
    SELECT 'Coche agregado exitosamente.';
  END IF;
END; //

CREATE PROCEDURE AgregarPuesto(
  IN $puesto VARCHAR(50),
  IN $salario DECIMAL(6,2)
)
BEGIN
  INSERT INTO puesto (puesto, salario)
  VALUES ($puesto, $salario);
  SELECT 'Puesto agregado exitosamente.';
END; //
```

```
CREATE PROCEDURE AgregarDireccion(
  IN $estado INT,
  IN $codigo_postal INT,
  IN $calle VARCHAR(250),
  IN $numero_exterior INT,
  IN $numero_interior INT
)
BEGIN
  INSERT INTO direccion (id_estado, codigo_postal, calle, numero_exterior, numero_interior)
  VALUES ($estado, $codigo_postal, $calle, $numero_exterior, IFNULL($numero_interior, NULL));
  SELECT 'Dirección agregada exitosamente.';
END; //

CREATE PROCEDURE AgregarPaquete(
  IN $promocion VARCHAR(50),
  IN $descripcion VARCHAR(250),
  IN $precio DECIMAL(6,2)
)
BEGIN
  INSERT INTO paquete (promocion, descripcion, precio)
  VALUES ($promocion, $descripcion, $precio);
  SELECT 'Paquete agregado exitosamente.';
END; //

CREATE PROCEDURE GenerarTicket(
  IN $cliente CHAR(18),
  IN $operador INT,
  IN $coche CHAR(8),
  IN $sucursal INT,
  IN $tipo_pago INT,
  IN $paquete INT,
  IN $promocion INT,
  IN $comentario VARCHAR(250),
```

```
  IN $subtotal DECIMAL(8,2),
  IN $total DECIMAL(8,2)
)
BEGIN
  DECLARE $existe INT DEFAULT 0;
  DECLARE $ultimo INT;
  SELECT COUNT(*) INTO $existe
  FROM ticket
  WHERE cliente = $cliente AND coche = $coche AND sucursal = $sucursal;
  IF $existe > 0 THEN
    SELECT 'El ticket ya existe.';
  ELSE
    START TRANSACTION;
    INSERT INTO ticket (cliente, operador, coche, sucursal, tipo_pago, paquete, promocion, comentario, subtotal, total)
    VALUES ($cliente, $operador, $coche, $sucursal, $tipo_pago, $paquete, IFNULL($promocion, NULL), IFNULL($comentario, ''), $subtotal, $total);
    SET $ultimo = LAST_INSERT_ID();
    COMMIT;
    SELECT 'Ticket generado exitosamente.';
    SELECT * FROM ticket WHERE id_ticket = $ultimo;
  END IF;
END; //

CREATE PROCEDURE AgregarEstado(
  IN nombre VARCHAR(20)
)
BEGIN
  INSERT INTO estado (estado)
  VALUES (nombre);
  SELECT 'Esdtado agregado existosamente.';
END; //
```

```
CALL AgregarEstado('Ciudad de México');
CALL AgregarEstado('Jalisco');
CALL AgregarEstado('Puebla');
CALL AgregarEstado('Oaxaca');
CALL AgregarEstado('Quintana Roo');

CALL AgregarDireccion(1, 72000, 'Calle de los Ángeles', 101, 201);
CALL AgregarDireccion(2, 44100, 'Avenida Hidalgo', 102, NULL);
CALL AgregarDireccion(3, 68000, 'Boulevard Benito Juárez', 103, NULL);
CALL AgregarDireccion(4, 77500, 'Calle Quintana Roo', 104, 204);
CALL AgregarDireccion(5, 11560, 'Paseo de la Reforma', 105, NULL);

CALL AgregarSucursal('Sucursal Centro', 1, 1);
CALL AgregarSucursal('Sucursal Norte', 2, 2);
CALL AgregarSucursal('Sucursal Sur', 3, 3);
CALL AgregarSucursal('Sucursal Este', 4, 4);
CALL AgregarSucursal('Sucursal Oeste', 5, 5);

CALL AgregarHorario(TIME('08:00:00'), TIME('18:00:00'));
CALL AgregarHorario(TIME('09:00:00'), TIME('19:00:00'));

CALL AgregarPuesto('Gerente General', 2500.00);
CALL AgregarPuesto('Asistente de Gerencia', 1500.00);
CALL AgregarPuesto('Lavador de Autos', 800.00);
CALL AgregarPuesto('Contador Financiero', 2000.00);
CALL AgregarPuesto('Recepcionista de Servicio al Cliente', 1200.00);

CALL AgregarEmpleado('CURP001', 'Juan Pérez', 'Gerente General', 1, '555-123-4567',
'jperez@autowash.com');
CALL AgregarEmpleado('CURP002', 'María López', 'Asistente de Gerencia', 2, '555-234-5678',
'mlopez@autowash.com');
CALL AgregarEmpleado('CURP003', 'Carlos García', 'Lavador de Autos', 3, '555-345-6789',
'cgarcia@autowash.com');
```

```
CALL AgregarEmpleado('CURP004', 'Ana Martínez', 'Contador Financiero', 4, '555-456-7890', 'amartinez@autowash.com');
CALL AgregarEmpleado('CURP005', 'Luis Rodríguez', 'Recepcionista de Servicio al Cliente', 5, '555-567-8901', 'lrodriguez@autowash.com');

CALL AgregarPaquete('Lavado Básico', 'Incluye lavado exterior e interior básico.', 150.00);
CALL AgregarPaquete('Lavado Completo', 'Incluye lavado exterior e interior completo con cera.', 250.00);
CALL AgregarPaquete('Lavado Premium', 'Incluye lavado completo más pulido y encerado.', 350.00);
CALL AgregarPaquete('Lavado y Desinfección', 'Incluye lavado completo más desinfección interior.', 300.00);
CALL AgregarPaquete('Lavado Express', 'Lavado exterior rápido para clientes en movimiento.', 100.00);

CALL AgregarMembresia('Membresía Bronce', 'Descuento del 10% en todos los servicios.', 500.00);
CALL AgregarMembresia('Membresía Plata', 'Descuento del 15% en todos los servicios y un lavado gratis al mes.', 1000.00);
CALL AgregarMembresia('Membresía Oro', 'Descuento del 20% en todos los servicios y dos lavados gratis al mes.', 1500.00);

CALL AgregarPromocion('Promoción de Verano', '20% de descuento en lavado completo.', NOW(), DATE_ADD(NOW(), INTERVAL 3 MONTH));
CALL AgregarPromocion('Promoción de Invierno', 'Lavado y desinfección con precio especial.', NOW(), DATE_ADD(NOW(), INTERVAL 3 MONTH));

CALL AgregarCliente('CURP006','Laura Jiménez','laurajimenez@genial.com','555-678-9012');
CALL AgregarCliente('CURP007','Oscar Hernández','oscarhernandez@yahoo1.com','555-789-0123');

CALL AgregarCoche('MEX1234', 'CURP001', 'Toyota Corolla', 2021, 'Gris');
CALL AgregarCoche('MEX5678', 'CURP002', 'Honda Civic', 2020, 'Negro');
CALL AgregarCoche('MEX9101', 'CURP003', 'Ford Focus', 2019, 'Blanco');
CALL AgregarCoche('MEX1121', 'CURP004', 'Nissan Sentra', 2018, 'Azul');
```

```
CALL AgregarCoche('MEX3141', 'CURP005', 'Chevrolet Aveo', 2017, 'Rojo');


CALL GenerarTicket('CURP001', 1, 'MEX1234', 1, 1, 1, NULL, 'Me gustó', 150.00, 180.00);
CALL GenerarTicket('CURP002', 2, 'MEX5678', 2, 2, 2, NULL, 'Dejan bien sucio', 250.00, 300.00);
CALL GenerarTicket('CURP003', 3, 'MEX9101', 3, 3, NULL, NULL,'Comentario esperando revision' ,350.00 ,420.00 );
CALL GenerarTicket('CURP004' ,4 ,'MEX1121' ,4 ,4 ,NULL ,NULL ,'Los baños bien cerdos' ,300.00 ,360.00 );
CALL GenerarTicket('CURP005' ,5 ,'MEX3141' ,5 ,5 ,NULL ,NULL ,'Hola miss iliana' ,100.00 ,120.00 );
```