```
/*
 Prueba Técnica: Base de Datos y Lógica de Programación
 Sección 1: Evaluación de Bases de Datos SQL

 Ruth de León - 2024
*/

--Parte 1: Creación de la base de datos
CREATE DATABASE PruebaTecnica;

--Parte 2: Tablas
CREATE TABLE Clientes (
   ClienteId SERIAL PRIMARY KEY,
   Nombre VARCHAR(100) NOT NULL,
   Correo VARCHAR(100) UNIQUE NOT NULL
);

CREATE TABLE Productos (
   ProductoId SERIAL PRIMARY KEY,
   Nombre VARCHAR(100) NOT NULL,
   Precio DECIMAL(10, 2) NOT NULL,
   Cantidad INT NOT NULL
);

CREATE TABLE Ordenes (
   OrdenId SERIAL PRIMARY KEY,
   ClienteId INT,
   Fecha DATE NOT NULL,
   FOREIGN KEY (ClienteId) REFERENCES Clientes(ClienteId)
```

```
);

CREATE TABLE DetalleOrden (
  DetalleId SERIAL PRIMARY KEY,
  OrdenId INT,
  ProductoId INT,
  Cantidad INT NOT NULL,
  PrecioUnitario DECIMAL(10, 2) NOT NULL,
  FOREIGN KEY (OrdenId) REFERENCES Ordenes(OrdenId),
  FOREIGN KEY (ProductoId) REFERENCES Productos(ProductoId)
);


--Parte 3: Operaciones CRUD

-----Insertar
INSERT INTO Productos (Nombre, Precio, Cantidad)
VALUES ('Pan integral', 30.99, 55);


-----Actualizar
UPDATE Productos
SET Precio = 45.10
WHERE ProductoId = 1;

select * from Productos;

-----Eliminar
DELETE FROM Productos
```

```
WHERE ProductoId = 1;


-----Seleccionar
SELECT * FROM Productos;



--Parte 4: Procedimientos Almacenados (SP)

-----sp_ObtenerProductoDetalles
CREATE PROCEDURE sp_ObtenerProductoDetalles(IN ProductoId INT)
BEGIN
  SELECT Nombre, Precio, Cantidad, (Precio * Cantidad) AS ValorTotal
  FROM Productos
  WHERE ProductoId = ProductoId;
END;

-----sp_ObtenerOrdenesPorCliente
CREATE PROCEDURE sp_ObtenerOrdenesPorCliente(IN ClienteId INT)
BEGIN
  SELECT o.OrdenId, o.Fecha, SUM(d.Cantidad * d.PrecioUnitario) AS TotalOrden
  FROM Ordenes o
  JOIN DetalleOrden d ON o.OrdenId = d.OrdenId
  WHERE o.ClienteId = ClienteId
  GROUP BY o.OrdenId, o.Fecha;
END;
```

-----sp_ObtenerDetalleOrdenCompleto

```
CREATE PROCEDURE sp_ObtenerDetalleOrdenCompleto(IN OrdenId INT)
BEGIN
  SELECT c.Nombre, o.Fecha, p.Nombre AS Producto, d.Cantidad, d.PrecioUnitario,
      (d.Cantidad * d.PrecioUnitario) AS TotalDetalle
  FROM Ordenes o
  JOIN Clientes c ON o.ClienteId = c.ClienteId
  JOIN DetalleOrden d ON o.OrdenId = d.OrdenId
  JOIN Productos p ON d.ProductoId = p.ProductoId
  WHERE o.OrdenId = OrdenId;
END;
```

-----sp_ObtenerTotalVentas

```
CREATE PROCEDURE sp_ObtenerTotalVentas(IN FechaInicio DATE, IN FechaFin DATE)
BEGIN
  SELECT SUM(d.Cantidad * d.PrecioUnitario) AS TotalVentas
  FROM DetalleOrden d
  JOIN Ordenes o ON d.OrdenId = o.OrdenId
  WHERE o.Fecha BETWEEN FechaInicio AND FechaFin;
END;
```

--Parte 5: Trigger

----trg_ActualizarCantida

```
CREATE TRIGGER trg_ActualizarCantidad
```

```
AFTER UPDATE ON Productos
FOR EACH ROW
BEGIN
  INSERT INTO Historial (ProductoId, CantidadAntigua, CantidadNueva, FechaCambio)
  VALUES (OLD.ProductoId, OLD.Cantidad, NEW.Cantidad, NOW());
END;
```

```
--Parte 6: Función
CREATE FUNCTION fn_ObtenerPrecioPromedio()
RETURNS DECIMAL(10, 2)
BEGIN
  DECLARE Promedio DECIMAL(10, 2);
  SELECT AVG(Precio) INTO Promedio FROM Productos;
  RETURN Promedio;
END;
```

```
--Parte 7: Relación Master-Detalle y Consultas

----Insertar una nueva orden y sus detalles
```

```
INSERT INTO Clientes (ClienteId, Nombre, Correo)
VALUES (1, 'Anaí Morataya', 'Ann_Mora@yahoo.com');

INSERT INTO Productos (ProductoId, Nombre, Precio, Cantidad)
VALUES (1, 'Mayonesa', 10.01, 10);

INSERT INTO Productos (ProductoId, Nombre, Precio, Cantidad)
VALUES (6, 'Sopa', 2.50, 110);

INSERT INTO Clientes (ClienteId, Nombre, Correo)
VALUES (2, 'Sara Lopez', 'saraL@gmail.com');


WITH new_order AS (
  INSERT INTO Ordenes (ClienteId, Fecha)
  VALUES (2, CURRENT_DATE)
  RETURNING OrdenId
)
INSERT INTO DetalleOrden (OrdenId, ProductoId, Cantidad, PrecioUnitario)
SELECT OrdenId, 6, 0, 10
FROM new_order;


----Seleccionar todas las órdenes con detalles de productos
SELECT o.OrdenId, o.Fecha, p.Nombre AS Producto, d.Cantidad, d.PrecioUnitario
FROM Ordenes o
LEFT JOIN DetalleOrden d ON o.OrdenId = d.OrdenId
LEFT JOIN Productos p ON d.ProductoId = p.ProductoId;
```

----Seleccionar todas las órdenes y sus detalles, incluyendo las órdenes sin detalles

SELECT *o*.***OrdenId***, *o*.***Fecha***, *p*.***Nombre*** **AS** *Producto*, *d*.***Cantidad***, *d*.***PrecioUnitario***

FROM ***Ordenes*** *o*

LEFT JOIN ***DetalleOrden*** *d* **ON** *o*.***OrdenId*** = *d*.***OrdenId***

LEFT JOIN ***Productos*** *p* **ON** *d*.***ProductoId*** = *p*.***ProductoId***;

----Seleccionar todos los productos y sus detalles de órdenes, incluyendo productos sin órdenes

SELECT *p*.***Nombre***, *d*.***Cantidad***, *d*.***PrecioUnitario***

FROM ***Productos*** *p*

LEFT JOIN ***DetalleOrden*** *d* **ON** *p*.***ProductoId*** = *d*.***ProductoId***;

----Seleccionar todas las órdenes con detalles de productos y la información del cliente

SELECT *o*.***OrdenId***, *o*.***Fecha***, *c*.***Nombre*** **AS** *Cliente*, *p*.***Nombre*** **AS** *Producto*, *d*.***Cantidad***, *d*.***PrecioUnitario***

FROM ***Ordenes*** *o*

JOIN ***Clientes*** *c* **ON** *o*.***ClienteId*** = *c*.***ClienteId***

LEFT JOIN ***DetalleOrden*** *d* **ON** *o*.***OrdenId*** = *d*.***OrdenId***

LEFT JOIN ***Productos*** *p* **ON** *d*.***ProductoId*** = *p*.***ProductoId***;