Taller 3: Normalización de Bases de datos - Forma Normal

Camilo Esteban Paez - 1110450891

Problemas Identificados

La tabla productos tiene las siguiente columnas: PedidoID | ClienteID | ProductoID | Cantidad | Fecha | MetodoPago , PedidoID es su PRIMARY KEY, ClientID | ProductoID son sus FOREIGN KEYS que permiten la no repetición de la información tanto de los clientes como del producto, Cantidad es un INT lo cual es tal-vez algo excesivo, Fecha es un DATE y MetodoPago es un VARCHAR(50) aquí también hay un problema.

No relacionados a la normalización

• Cantidad tiene tipo INT firmado, al ser una tienda minorista no es necesario una cantidad posible tan grande, ademas no es posible pedir una cantidad negativa

Relacionados con la normalización

- MetodoPago se comporta como un Enum pero esta definido como string, se puede extraer a otra tabla, compuesta por las columnas MetodoPagoId | descripcion o en su defecto usar el datatype Enum.
- Categoria se comporta como un Enum pero esta definido como string, se puede extraer a otra tabla, compuesta por las columnas Categoria | descripcion o en su defecto usar el datatype Enum.
- · Direcciones podria extraerse a una tabla .

Cambios realizados y nueva estructura

Atomizacion

La base de datos esta suficientemente atomizada, se podria separar Nombre aunque no lo considero necesario.

Eliminar valores multivaluados y repetidos

Creacion tabla Categoria

Categoria pasara a residir en una tabla donde contará con una ID y una descripcion, esto permitira que se evite redundancia en la tabla Productos

```
create table Categoria (
   CategoriaID INT AUTO_INCREMENT PRIMARY KEY,
   descripcion VARCHAR(50)
);
```

· Creacion tabla MetodoPago

MetodoPago pasara a residir en una tabla donde contará con una ID y una descripcion, esto permitira que se evite redundancia en la tabla MetodoPago

```
create table MetodoPago (
   MetodoPagoID INT AUTO_INCREMENT PRIMARY KEY,
   descripcion VARCHAR(50)
);
```

Nueva tabla Productos

```
create table Productos (
   ProductoID INT AUTO_INCREMENT PRIMARY KEY,
   Nombre VARCHAR(100),
   CategoriaID INT, -- <---- Nuevo Tipo
   Precio DECIMAL(10, 2),
   FOREIGN KEY (CategoriaID) REFERENCES Categoria(CategoriaID) -- <----
Nueva Foreign Key
);</pre>
```

· Nueva tabla Pedidos

```
create table Pedidos (
    PedidoID INT AUTO_INCREMENT PRIMARY KEY,
    ClienteID INT,
    ProductoID INT,
    Cantidad INT,
    Fecha DATE,
    MetodoPagoID INT, -- <---- Nuevo Tipo
    FOREIGN KEY (ClienteID) REFERENCES Clientes(ClienteID),
    FOREIGN KEY (ProductoID) REFERENCES Productos(ProductoID),
    FOREIGN KEY (MetodoPagoID) REFERENCES MetodoPago(MetodoPagoID) -- <----
    Nueva Foreign Key
);</pre>
```

Clientes se mantiene con su misma estructura

```
create table Clientes (
    ClienteID INT AUTO_INCREMENT PRIMARY KEY,
    Nombre VARCHAR(100),
    Direccion VARCHAR(150),
    Telefono VARCHAR(15)
);
```

Capturas

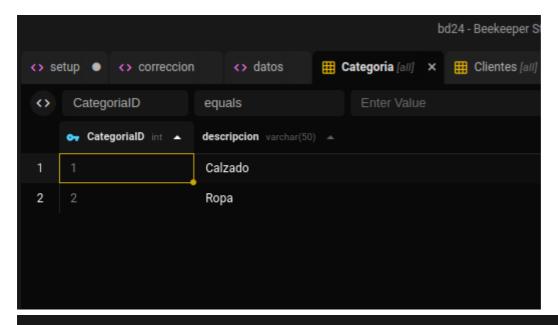
```
setup • <> correccion × <> datos
                                         Categoria [all]
                                                            Clientes [all]
                                                                              Productos [all]
    create table Categoria (
      CategoriaID INT AUTO_INCREMENT PRIMARY KEY,
      descripcion VARCHAR(50)
    create table Clientes (
         ClienteID INT AUTO_INCREMENT PRIMARY KEY,
         Nombre VARCHAR(100),
         Direccion VARCHAR(150),
         Telefono VARCHAR(15)
    );
    create table MetodoPago (
      MetodoPagoID INT AUTO_INCREMENT PRIMARY KEY,
      descripcion VARCHAR(50)
    create table Productos (
      ProductoID INT AUTO_INCREMENT PRIMARY KEY,
      Nombre VARCHAR(100),
      CategoriaID INT,
      Precio DECIMAL(10, 2),
      FOREIGN KEY (CategoriaID) REFERENCES Categoria(CategoriaID)
    create table Pedidos (
         PedidoID INT AUTO_INCREMENT PRIMARY KEY,
         ClienteID INT,
         ProductoID INT,
         Cantidad INT,
         Fecha DATE,
         MetodoPagoID INT,
         FOREIGN KEY (ClienteID) REFERENCES Clientes(ClienteID),
         FOREIGN KEY (ProductoID) REFERENCES Productos(ProductoID),
         FOREIGN KEY (MetodoPagoID) REFERENCES MetodoPago(MetodoPagoID)
    );

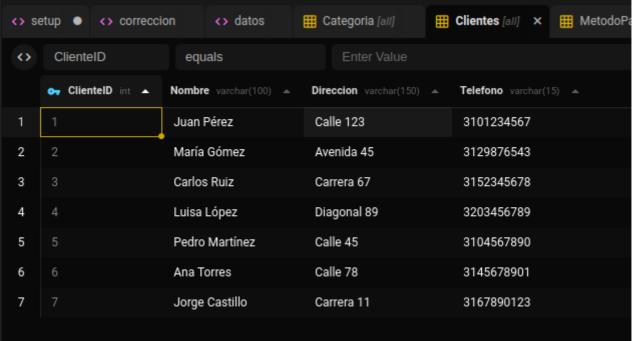
← datos ★ III Categoria [all]
                                                          Clientes [all]

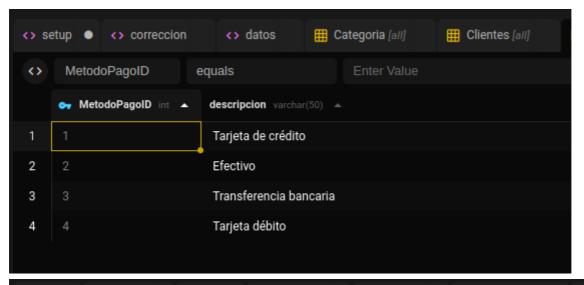
    MetodoPago [all]

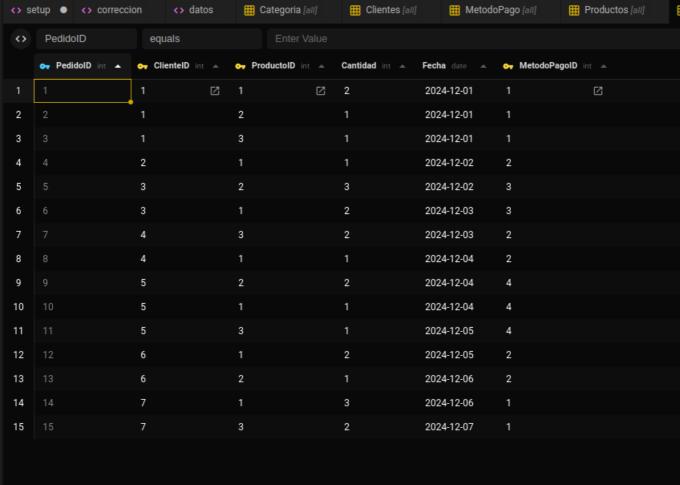
                                                                                                Productos [all]
                                                                                                                   # Pedid
    INSERT INTO Clientes (Nombre, Direccion, Telefono) VALUES ('Juan Pérez', 'Calle 123', '3101234567'), ('María Gómez', 'Avenida 45', '3129876543'), ('Carlos Ruiz', 'Carrera 67', '3152345678'), ('Luisa López', 'Diagonal 89', '3203456789'), ('Pedro Martínez', 'Calle 45', '3104567890'),
    ('Ana Torres', 'Calle 78', '3145678901'),
    ('Jorge Castillo', 'Carrera 11', '3167890123');
    insert into Categoria(descripcion) values
    ('Calzado'),
    ('Ropa');
    INSERT INTO Productos (Nombre, CategoriaID, Precio) VALUES
    ('Zapatos', 1, 60.00),
    ('Camisa', 2, 30.00),
('Pantalón', 2, 50.00);
    insert\ into\ MetodoPago(descripcion)\ VALUES
    ('Tarjeta de crédito'),
    ('Efectivo'),
    ('Transferencia bancaria'),
    ('Tarjeta débito');
    insert into Pedidos (ClienteID, ProductoID, Cantidad, Fecha, MetodoPagoID) VALUES
    (1, 1, 2, '2024-12-01', 1),
    (1, 2, 1, '2024-12-01', 1),
(1, 3, 1, '2024-12-01', 1),
    (2, 1, 1, '2024-12-02', 2),
```

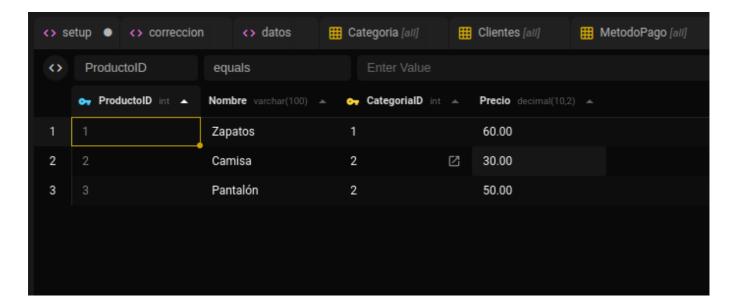
```
30 (3, 2, 3, '2024-12-02', 3),
31 (3, 1, 2, '2024-12-03', 3),
32 (4, 3, 2, '2024-12-03', 2),
33 (4, 1, 1, '2024-12-04', 2),
34 (5, 2, 2, '2024-12-04', 4),
35 (5, 1, 1, '2024-12-04', 4),
36 (5, 3, 1, '2024-12-05', 4),
37 (6, 1, 2, '2024-12-05', 2),
38 (6, 2, 1, '2024-12-06', 2),
39 (7, 1, 3, '2024-12-06', 1),
40 (7, 3, 2, '2024-12-07', 1);
```











Conclusiones

La normalizacion es esencial en las bases de datos relacionales ya que sacan partido de las propias caracteristicas de esta, permiten que ademas de eficiente se mantenga mantenible a lo largo del tiempo.