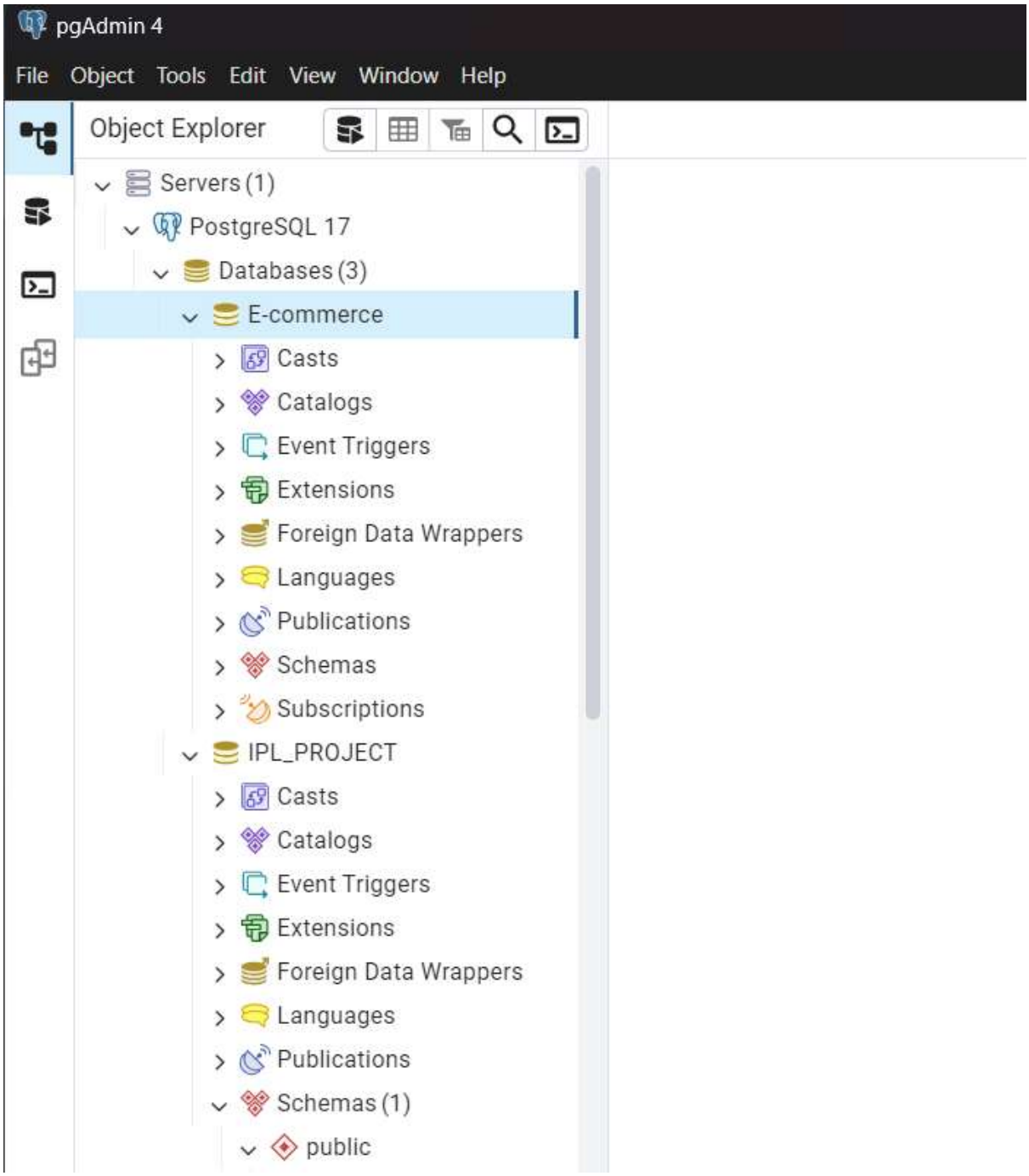
1. Learn to create databases, tables, and define relationships.

Ans. Create database on the postgre sql or whatever you want to use

```
CREATE DATABASE ecommerce_db;
\c ecommerce_db  -- if using psql
-- OR
Switch to the new database:
USE ecommerce_db; -- if MySQL

CREATE TABLE Customer (
    CustomerID SERIAL PRIMARY KEY,
    Name VARCHAR(100),
    Email VARCHAR(100) UNIQUE,
    Address TEXT
);

CREATE TABLE Category (
    CategoryID SERIAL PRIMARY KEY,
    Name VARCHAR(100)
);

CREATE TABLE Product (
    ProductID SERIAL PRIMARY KEY,
    Name VARCHAR(100),
    Price DECIMAL(10, 2),
    CategoryID INT,
    FOREIGN KEY (CategoryID) REFERENCES Category(CategoryID)
);

CREATE TABLE Orders (
    OrderID SERIAL PRIMARY KEY,
```

```
    CustomerID INT,
    OrderDate DATE,
    FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)
);

CREATE TABLE OrderItem (
    OrderID INT,
    ProductID INT,
    Quantity INT,
    PRIMARY KEY (OrderID, ProductID),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Product(ProductID)
);

CREATE TABLE Payment (
    PaymentID SERIAL PRIMARY KEY,
    OrderID INT,
    PaymentDate DATE,
    Amount DECIMAL(10,2),
    PaymentMethod VARCHAR(50),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID)
);

CREATE TABLE Shipping (
    ShippingID SERIAL PRIMARY KEY,
    OrderID INT,
    ShippingDate DATE,
    ShippingAddress TEXT,
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID)
);
```

2.Tool (postgre sql)



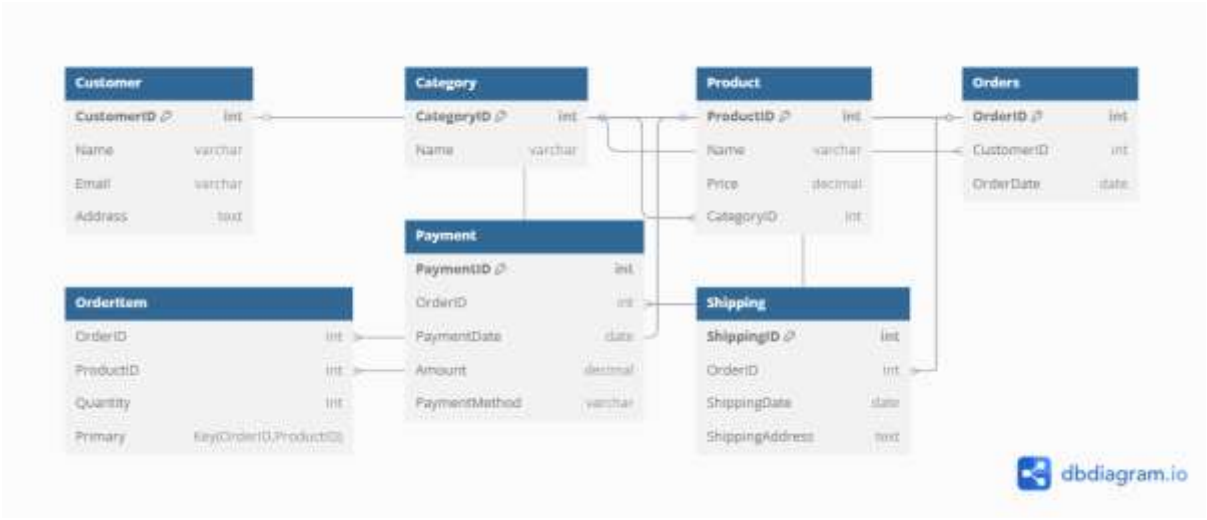
3. Deliverables: SQL script to create schema and ER diagram.

-- Create database

CREATE DATABASE ecommerce_db;

```
-- Switch to the new database (for CLI tools)
-- \c ecommerce_db

-- Customer Table
CREATE TABLE Customer (
    CustomerID SERIAL PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Email VARCHAR(100) UNIQUE NOT NULL,
    Address TEXT
);

-- Category Table
CREATE TABLE Category (
    CategoryID SERIAL PRIMARY KEY,
    Name VARCHAR(100) NOT NULL
);

-- Product Table
CREATE TABLE Product (
    ProductID SERIAL PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Price DECIMAL(10, 2) NOT NULL,
    CategoryID INT REFERENCES Category(CategoryID)
);

-- Orders Table
CREATE TABLE Orders (
    OrderID SERIAL PRIMARY KEY,
    CustomerID INT REFERENCES Customer(CustomerID),
    OrderDate DATE DEFAULT CURRENT_DATE
```

```
);

-- OrderItem Table (Junction Table)
CREATE TABLE OrderItem (
    OrderID INT,
    ProductID INT,
    Quantity INT DEFAULT 1,
    PRIMARY KEY (OrderID, ProductID),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Product(ProductID)
);

-- Payment Table
CREATE TABLE Payment (
    PaymentID SERIAL PRIMARY KEY,
    OrderID INT REFERENCES Orders(OrderID),
    PaymentDate DATE DEFAULT CURRENT_DATE,
    Amount DECIMAL(10, 2),
    PaymentMethod VARCHAR(50)
);

-- Shipping Table
CREATE TABLE Shipping (
    ShippingID SERIAL PRIMARY KEY,
    OrderID INT REFERENCES Orders(OrderID),
    ShippingDate DATE DEFAULT CURRENT_DATE,
    ShippingAddress TEXT
);
```


Customer
CustomerID int
Name varchar
Email varchar
Address text
Category
CategoryID int
Name varchar
Product
ProductID int
Name varchar
Price decimal
CategoryID int
Orders
OrderID int
CustomerID int
OrderDate date
Payment
PaymentID int
OrderID int
PaymentDate date
Amount decimal
PaymentMethod varchar
OrderItem
OrderID int
ProductID int
Quantity int
Primary Key(OrderID,ProductID)
Shipping
ShippingID int
OrderID int
ShippingDate date
ShippingAddress text
dbdiagram.io

ER diagram