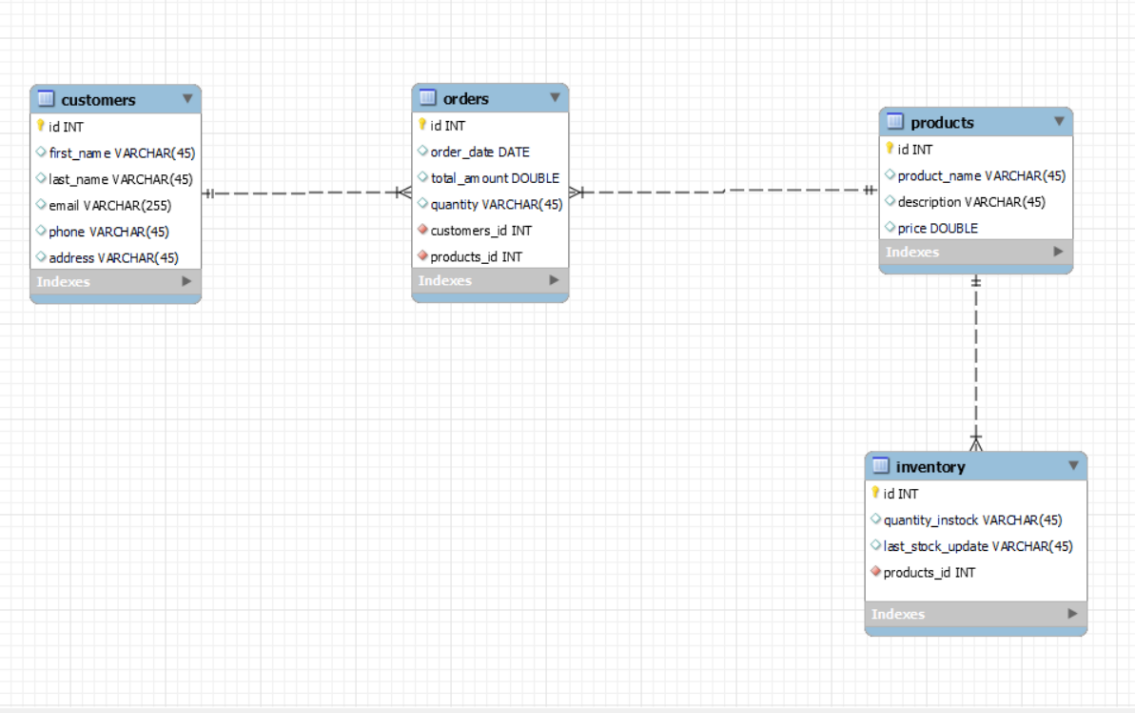
ASSIGNMENT-1 TechShop, an electronic gadgets shop

ER diagram:



CODE:

```
CREATE DATABASE TechShop;
USE TechShop;

CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Email VARCHAR(100),
    Phone VARCHAR(15),
```

```
    Address VARCHAR(255)
);

CREATE TABLE Products (
    ProductID INT PRIMARY KEY,
    ProductName VARCHAR(100),
    Description VARCHAR(255),
    Price DECIMAL(10, 2)
);

CREATE TABLE Orders (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    OrderDate DATE,
    TotalAmount DECIMAL(10, 2),
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
);

CREATE TABLE OrderDetails (
    OrderDetailID INT PRIMARY KEY,
    OrderID INT,
    ProductID INT,
    Quantity INT,
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
```

```
CREATE TABLE Inventory (
    InventoryID INT PRIMARY KEY,
    ProductID INT,
    QuantityInStock INT,
    LastStockUpdate DATE,
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);

INSERT INTO Customers VALUES
  (1, 'Sethu', 'Deepika', 'sethu.deepika@email.com', '1234567890',
'Andhrapradesh'),
  (2, 'Leena', 'Devi', 'leena.devi@email.com', '9876543210', 'Tamilnadu'),
  (3, 'Mani', 'Malar', 'mani.malar@email.com', '5551234567', 'Puducherry'),
  (4, 'Emily', 'Johnson', 'emily.j@email.com', '3216549870', 'chennai'),
  (5, 'Haritha', 'Karthikeyan', 'haritha.k@email.com', '7778889999', 'Himachal'),
  (6, 'Sophia', 'Miller', 'sophia.m@email.com', '4445556666', 'Madhyapradesh'),
  (7, 'Olivia', 'Davis', 'oliver.d@email.com', '1237894560', 'Hyderabad'),
  (8, 'Priyanka', 'Ramakrishna', 'priya.rama@email.com', '1112223333',
'Ahmadabad'),
  (9, 'Diya', 'Suraj', 'diya.suraj@email.com', '9998887777', 'Delhi'),
  (10, 'Avanthika', 'Jain', 'avanthika.jain@email.com', '5557778888',
'Manglore');

INSERT INTO Products VALUES
  (1, 'Laptop', 'High-performance laptop', 999.99),
  (2, 'Smartphone', 'Latest smartphone model', 699.99),
```

```
  (3, 'Tablet', 'Lightweight tablet', 349.99),
  (4, 'Headphones', 'Noise-canceling headphones', 149.99),
  (5, 'Camera', 'Professional-grade camera', 1299.99),
  (6, 'Smartwatch', 'Fitness tracking smartwatch', 199.99),
  (7, 'Speaker', 'Wireless Bluetooth speaker', 79.99),
  (8, 'Gaming Console', 'Next-gen gaming console', 499.99),
  (9, 'Printer', 'All-in-one printer', 199.99),
  (10, 'Monitor', 'Ultra HD computer monitor', 399.99);


INSERT INTO Orders VALUES
  (1, 1, '2024-03-01', 999.99),
  (2, 3, '2024-03-02', 699.99),
  (3, 5, '2024-03-03', 149.99),
  (4, 2, '2024-03-04', 1299.99),
  (5, 7, '2024-03-05', 199.99),
  (6, 9, '2024-03-06', 79.99),
  (7, 4, '2024-03-07', 499.99),
  (8, 6, '2024-03-08', 199.99),
  (9, 8, '2024-03-09', 399.99),
  (10, 10, '2024-03-10', 349.99);

INSERT INTO OrderDetails VALUES
  (1, 1, 1, 2),
  (2, 2, 3, 1),
  (3, 3, 4, 3),
```

```
  (4, 4, 6, 1),
  (5, 5, 7, 2),
  (6, 6, 8, 1),
  (7, 7, 10, 1),
  (8, 8, 2, 2),
  (9, 9, 5, 1),
  (10, 10, 9, 2);


INSERT INTO Inventory VALUES
  (1, 1, 20, '2024-03-01'),
  (2, 2, 15, '2024-03-02'),
  (3, 3, 30, '2024-03-03'),
  (4, 4, 10, '2024-03-04'),
  (5, 5, 25, '2024-03-05'),
  (6, 6, 18, '2024-03-06'),
  (7, 7, 22, '2024-03-07'),
  (8, 8, 12, '2024-03-08'),
  (9, 9, 8, '2024-03-09'),
  (10, 10, 40, '2024-03-10');

 -- Task 2: Select, Where, Between, AND, LIKE:

-- 1. Retrieve the names and emails of all customers:
SELECT FirstName, LastName, Email FROM Customers;
```

```
-- 2. List all orders with their order dates and corresponding customer names:

SELECT o.OrderID, o.OrderDate, c.FirstName, c.LastName

FROM Orders o

JOIN Customers c ON o.CustomerID = c.CustomerID;


-- 3. Insert a new customer record into the "Customers" table:

INSERT INTO Customers ( CustomerID,FirstName, LastName, Email, Phone,
Address)

VALUES (11,'New', 'Customer', 'new.customer@email.com', '1234567890',
'New Address');


-- 4. Update the prices of all electronic gadgets by increasing them by 10%:

-- Modify the data type of the 'Price' column to accommodate larger values

ALTER TABLE Products

MODIFY COLUMN Price DECIMAL(12, 2);



-- Update the prices of all electronic gadgets by increasing them by 10%

UPDATE Products

SET Price = ROUND(Price * 1.10, 2)

WHERE ProductID IN (1, 2, 3, 4, 5, 6, 7, 8, 9, 10);



-- 5. Delete a specific order and its associated order details:

DELETE FROM OrderDetails WHERE OrderID = 5;
```

```
DELETE FROM Orders WHERE OrderID = 5;

-- 6. Insert a new order into the "Orders" table:
INSERT INTO Orders (OrderId,CustomerID, OrderDate, TotalAmount)
VALUES (13,10, '2024-03-15', 1200.00);

-- 7. Update the contact information of a specific customer:
UPDATE Customers
SET Email = 'mitali.b@email.com', Address = 'Banglore'
WHERE CustomerID = 10;

-- 8. Recalculate and update the total cost of each order:
-- Update the total cost of each order in the "Orders" table
UPDATE Orders
SET TotalAmount = (
    SELECT SUM(od.Quantity * p.Price)
    FROM OrderDetails od
    JOIN Products p ON od.ProductID = p.ProductID
    WHERE od.OrderID = Orders.OrderID
)
WHERE OrderID IN (1, 2, 3, 4, 5, 6, 7, 8, 9, 10);


-- 9. Delete all orders and their associated order details for a specific customer:
SET SQL_SAFE_UPDATES = 0;
DELETE FROM OrderDetails WHERE OrderID IN (SELECT OrderID FROM Orders
WHERE CustomerID = 7);
```

```
DELETE FROM Orders WHERE CustomerID = 7;

-- 10. Insert a new electronic gadget product into the "Products" table:

ALTER TABLE Products ADD COLUMN Category VARCHAR(100);

INSERT INTO Products (ProductID, ProductName, Description, Price, Category)
VALUES
  (11, 'Wireless Earbuds', 'High-quality wireless earbuds', 89.99, 'Audio Accessories');

-- 11. Update the status of a specific order:
ALTER TABLE Orders
ADD COLUMN Status VARCHAR(50);

UPDATE Orders SET Status = 'Shipped' WHERE OrderID = 8;

-- 12. Calculate and update the number of orders placed by each customer:
ALTER TABLE Customers
ADD COLUMN TotalOrders INT;
UPDATE Customers c
SET TotalOrders = (
  SELECT COUNT(*)
  FROM Orders o
  WHERE o.CustomerID = c.CustomerID
);
```

-- TASK 3

-- 1. Retrieve a list of all orders along with customer information:

```
SELECT Orders.OrderID, Customers.CustomerID, Customers.FirstName,
Customers.LastName, Customers.Email, Customers.Phone

FROM Orders

JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

-- 2. Find the total revenue generated by each electronic gadget product:

```
SELECT Products.ProductID, Products.ProductName,
SUM(OrderDetails.Quantity * Products.Price) AS TotalRevenue

FROM Products

JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID

GROUP BY Products.ProductID, Products.ProductName;
```

-- 3. List all customers who have made at least one purchase:

```
SELECT DISTINCT Customers.CustomerID, Customers.FirstName,
Customers.LastName, Customers.Email, Customers.Phone

FROM Customers

JOIN Orders ON Customers.CustomerID = Orders.CustomerID;
```

-- 4. Find the most popular electronic gadget (highest total quantity ordered):

```
SELECT Products.ProductID, Products.ProductName,
SUM(OrderDetails.Quantity) AS TotalQuantityOrdered

FROM Products
```

```
JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID
GROUP BY Products.ProductID, Products.ProductName
ORDER BY TotalQuantityOrdered DESC
LIMIT 1;

-- 5. Retrieve a list of electronic gadgets along with their corresponding categories:
CREATE TABLE Categories (
    CategoryID INT PRIMARY KEY,
    CategoryName VARCHAR(100)
);

INSERT INTO Categories VALUES
    (1, 'Electronics'),
    (2, 'Gadgets');

ALTER TABLE Products
ADD COLUMN CategoryID INT;

UPDATE Products
SET CategoryID = 1
WHERE ProductID IN (1, 2);

SELECT Products.ProductID, Products.ProductName, Categories.CategoryName
FROM Products
JOIN Categories ON Products.CategoryID = Categories.CategoryID;
```

```
-- 6. Calculate the average order value for each customer:

SELECT Customers.CustomerID, Customers.FirstName, Customers.LastName,
AVG(Orders.TotalAmount) AS AverageOrderValue

FROM Customers

JOIN Orders ON Customers.CustomerID = Orders.CustomerID

GROUP BY Customers.CustomerID, Customers.FirstName,
Customers.LastName;


-- 7. Find the order with the highest total revenue:

SELECT Orders.OrderID, Customers.CustomerID, Customers.FirstName,
Customers.LastName, Customers.Email, Customers.Phone,
SUM(OrderDetails.Quantity * Products.Price) AS TotalRevenue

FROM Orders

JOIN Customers ON Orders.CustomerID = Customers.CustomerID

JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

JOIN Products ON OrderDetails.ProductID = Products.ProductID

GROUP BY Orders.OrderID, Customers.CustomerID, Customers.FirstName,
Customers.LastName, Customers.Email, Customers.Phone

ORDER BY TotalRevenue DESC

LIMIT 1;


-- 8. List electronic gadgets and the number of times each product has been
ordered:

SELECT Products.ProductID, Products.ProductName,
COUNT(OrderDetails.OrderDetailID) AS OrderCount

FROM Products

LEFT JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID

GROUP BY Products.ProductID, Products.ProductName;
```

```
-- 9. Find customers who have purchased a specific electronic gadget product:

SELECT Customers.CustomerID, Customers.FirstName, Customers.LastName,
Customers.Email, Customers.Phone

FROM Customers

JOIN Orders ON Customers.CustomerID = Orders.CustomerID

JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

JOIN Products ON OrderDetails.ProductID = Products.ProductID

WHERE Products.ProductName = 'Laptop';


-- 10. Calculate the total revenue generated by all orders within a specific time
period:

SELECT SUM(TotalAmount) AS TotalRevenue

FROM Orders

WHERE OrderDate BETWEEN '2024-03-04' AND '2024-03-10';


-- TASK 4



-- 1. Find out which customers have not placed any orders:

SELECT CustomerID, FirstName, LastName, Email, Phone

FROM Customers

WHERE CustomerID NOT IN (SELECT DISTINCT CustomerID FROM Orders);


-- 2. Calculate the total number of products available for sale:

SELECT COUNT(ProductID) AS TotalProducts

FROM Products;
```

```
-- 3. Calculate the total revenue generated by TechShop:

SELECT SUM(TotalAmount) AS TotalRevenue

FROM Orders;


-- 4. Calculate the average quantity ordered for products in a specific category:

SELECT AVG(OrderDetails.Quantity) AS AvgQuantityOrdered

FROM OrderDetails

JOIN Products ON OrderDetails.ProductID = Products.ProductID

WHERE Products.CategoryID = (SELECT CategoryID FROM Categories WHERE
CategoryName = 'Electronics');


-- 5. Calculate the total revenue generated by a specific customer (customer ID
as a parameter):

SELECT SUM(TotalAmount) AS TotalRevenue

FROM Orders

WHERE CustomerID = 3;


-- 6. Find customers who have placed the most orders. List their names and the
number of orders:

SELECT Customers.CustomerID, Customers.FirstName, Customers.LastName,
COUNT(Orders.OrderID) AS OrderCount

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID

GROUP BY Customers.CustomerID, Customers.FirstName, Customers.LastName

ORDER BY OrderCount DESC

LIMIT 1;
```

-- 7. Find the most popular product category (highest total quantity ordered across all orders):

```
SELECT Categories.CategoryID, Categories.CategoryName,
SUM(OrderDetails.Quantity) AS TotalQuantityOrdered

FROM Categories

JOIN Products ON Categories.CategoryID = Products.CategoryID

JOIN OrderDetails ON Products.ProductID = OrderDetails.ProductID

GROUP BY Categories.CategoryID, Categories.CategoryName

ORDER BY TotalQuantityOrdered DESC

LIMIT 1;
```

-- 8. Find the customer who has spent the most money (highest total revenue) on electronic gadgets:

```
SELECT Customers.CustomerID, Customers.FirstName, Customers.LastName,
SUM(OrderDetails.Quantity * Products.Price) AS TotalSpending

FROM Customers

JOIN Orders ON Customers.CustomerID = Orders.CustomerID

JOIN OrderDetails ON Orders.OrderID = OrderDetails.OrderID

JOIN Products ON OrderDetails.ProductID = Products.ProductID

WHERE Products.CategoryID = (SELECT CategoryID FROM Categories WHERE
CategoryName = 'Electronics')

GROUP BY Customers.CustomerID, Customers.FirstName, Customers.LastName

ORDER BY TotalSpending DESC

LIMIT 1;
```

-- 9. Calculate the average order value for all customers:

```
SELECT AVG(TotalAmount) AS AvgOrderValue
```

```
FROM Orders;


-- 10. Find the total number of orders placed by each customer and list their
names along with the order count:

SELECT Customers.CustomerID, Customers.FirstName, Customers.LastName,
COUNT(Orders.OrderID) AS OrderCount

FROM Customers

LEFT JOIN Orders ON Customers.CustomerID = Orders.CustomerID

GROUP BY Customers.CustomerID, Customers.FirstName,
Customers.LastName;
```