Tech Shop Assignment

# **Creating Schema:**

## 1. Customers:

## • CustomerID (Primary Key)

## • FirstName

## • LastName

## • Email

## • Phone

## • Address

**create table Customers(**

**CustomerID int PRIMARY KEY NOT NULL,**

**FirstName varchar(20) NOT NULL,**

**LastName varchar(20) NOT NULL,**

**Email varchar(30),**

**Phone int,**

**Address varchar(20));**

## 2. Products:

## • ProductID (Primary Key)

## • ProductName

## • Description

## • Price

**create table Products(**

**ProductID int PRIMARY KEY NOT NULL,**

**ProductName varchar(50),**

**Description varchar(80),**

**Price DECIMAL(10, 2);**

## 3. Orders:

## • OrderID (Primary Key)

## • CustomerID (Foreign Key referencing Customers)

## • OrderDate

## • TotalAmount

**create table Orders(**

**OrderID int PRIMARY KEY NOT NULL,**

**CustomerID int FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID),**

**OrderDate DATE,**

**TotalAmount DECIMAL(10, 2));**

## 4. OrderDetails:

## • OrderDetailID (Primary Key)

## • OrderID (Foreign Key referencing Orders)

## • ProductID (Foreign Key referencing Products)

## • Quantity

**create table OrderDetails(**

**OrderDetailID int PRIMARY KEY NOT NULL,**

**OrderID int FOREIGN KEY (OrderID) references Orders(OrderID),**

**ProductID int foreign key(ProductID) references Products(ProductID),**

**Quantity int);**

## 5. Inventory

## • InventoryID (Primary Key)

## • ProductID (Foreign Key referencing Products)

## • QuantityInStock

## • LastStockUpdate

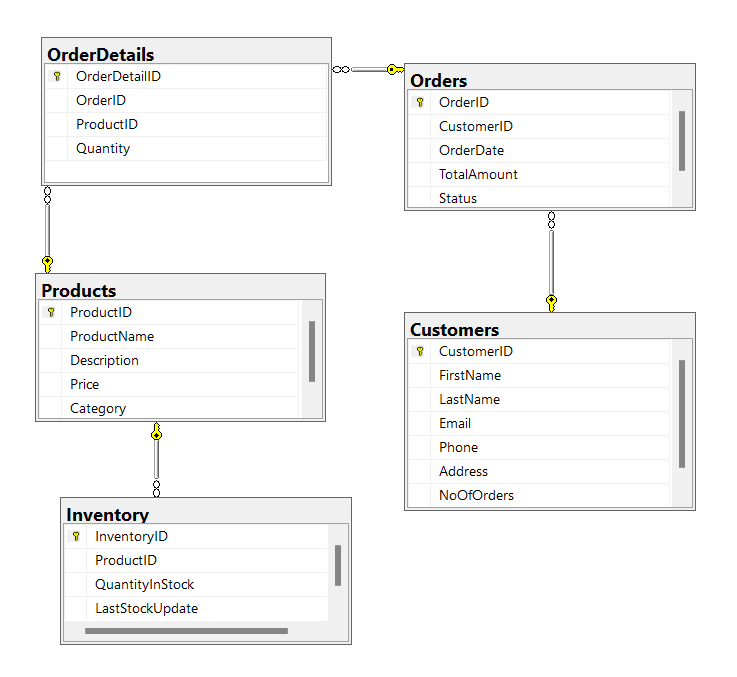
**create table Inventory(**

**InventoryID int PRIMARY KEY NOT NULL,**

**ProductID int foreign key (ProductID) references Products(ProductID),**

**QuantityInStock int,**

**LastStockUpdate int);**



# **Inserting Records:**

### **-- Insert 15 sample records into Customers table**

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

### **VALUES**

### **(1, 'John', 'Doe', 'john.doe@example.com', '123-456-7890', '123 Main St, Anytown, USA'),**

### **(2, 'Jane', 'Smith', 'jane.smith@example.com', '456-789-0123', '456 Elm St, Anycity, USA'),**

### **(3, 'Michael', 'Johnson', 'michael.johnson@example.com', '789-012-3456', '789 Oak St, Anystate, USA'),**

### **(4, 'Emily', 'Brown', 'emily.brown@example.com', '321-654-9870', '321 Maple St, Anyvillage, USA'),**

### **(5, 'Daniel', 'Martinez', 'daniel.martinez@example.com', '654-987-0123', '654 Cedar St, Anysuburb, USA'),**

### **(6, 'Sarah', 'Wilson', 'sarah.wilson@example.com', '987-012-3456', '987 Pine St, Anyhamlet, USA'),**

### **(7, 'David', 'Taylor', 'david.taylor@example.com', '111-222-3333', '111 Oak St, Anycity, USA'),**

### **(8, 'Olivia', 'Anderson', 'olivia.anderson@example.com', '444-555-6666', '444 Elm St, Anystate, USA'),**

### **(9, 'James', 'Hernandez', 'james.hernandez@example.com', '777-888-9999', '777 Maple St, Anytown, USA'),**

### **(10, 'Emma', 'Garcia', 'emma.garcia@example.com', '000-111-2222', '000 Cedar St, Anyvillage, USA'),**

### **(11, 'Ava', 'Lopez', 'ava.lopez@example.com', '333-444-5555', '333 Pine St, Anysuburb, USA'),**

### **(12, 'Alexander', 'Martinez', 'alexander.martinez@example.com', '666-777-8888', '666 Oak St, Anyhamlet, USA'),**

### **(13, 'Sophia', 'Gonzalez', 'sophia.gonzalez@example.com', '999-000-1111', '999 Elm St, Anytown, USA'),**

### **(14, 'Mia', 'Perez', 'mia.perez@example.com', '222-333-4444', '222 Maple St, Anycity, USA'),**

### **(15, 'Logan', 'Rodriguez', 'logan.rodriguez@example.com', '555-666-7777', '555 Cedar St, Anystate, USA');**

### **-- Insert 15 sample records into Products table**

### **INSERT INTO Products (ProductID, ProductName, Description, Price)**

### **VALUES**

### **(1, 'Smartphone', 'Smartphone with high-resolution camera', 599.99),**

### **(2, 'Laptop', 'Thin and lightweight laptop with SSD storage', 999.99),**

### **(3, 'Headphones', 'Wireless noise-canceling headphones', 199.99),**

### **(4, 'Smart Watch', 'Fitness tracker with heart rate monitor', 149.99),**

### **(5, 'Tablet', '10-inch tablet with retina display', 399.99),**

### **(6, 'Digital Camera', 'Mirrorless digital camera with 4K video recording', 799.99),**

### **(7, 'Gaming Console', 'Next-gen gaming console with VR support', 499.99),**

### **(8, 'Bluetooth Speaker', 'Portable Bluetooth speaker with long battery life', 79.99),**

### **(9, 'External Hard Drive', '1TB external hard drive with USB 3.0', 69.99),**

### **(10, 'Wireless Router', 'Dual-band wireless router for high-speed internet', 129.99),**

### **(11, 'Fitness Tracker', 'Waterproof fitness tracker with GPS', 129.99),**

### **(12, 'Smart Home Hub', 'Voice-controlled smart home hub', 149.99),**

### **(13, 'Wireless Earbuds', 'True wireless earbuds with touch controls', 129.99),**

### **(14, 'Monitor', '27-inch 4K monitor with IPS display', 399.99),**

### **(15, 'Printer', 'All-in-one printer with wireless connectivity', 199.99);**

### **-- Insert 15 sample records into Orders table**

### **INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)**

### **VALUES**

### **(1, 1, '2024-02-01', 249.99),**

### **(2, 3, '2024-02-03', 799.99),**

### **(3, 5, '2024-02-05', 149.99),**

### **(4, 2, '2024-02-07', 399.99),**

### **(5, 4, '2024-02-10', 999.99),**

### **(6, 6, '2024-02-12', 79.99),**

### **(7, 8, '2024-02-15', 129.99),**

### **(8, 10, '2024-02-18', 499.99),**

### **(9, 12, '2024-02-20', 129.99),**

### **(10, 14, '2024-02-22', 69.99),**

### **(11, 7, '2024-02-25', 149.99),**

### **(12, 9, '2024-02-28', 129.99),**

### **(13, 11, '2024-03-01', 399.99),**

### **(14, 13, '2024-03-03', 199.99),**

### **(15, 15, '2024-03-05', 999.99);**

### **INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)**

### **VALUES**

### **(16, 1, '2024-03-08', 329.99),**

### **(17, 3, '2024-03-10', 699.99),**

### **(18, 5, '2024-03-12', 199.99),**

### **(19, 2, '2024-03-15', 449.99),**

### **(20, 4, '2024-03-18', 1299.99),**

### **(21, 6, '2024-03-20', 89.99),**

### **(22, 8, '2024-03-22', 149.99),**

### **(23, 10, '2024-03-25', 599.99),**

### **(24, 12, '2024-03-28', 179.99),**

### **(25, 14, '2024-03-30', 79.99),**

### **(26, 7, '2024-04-01', 199.99),**

### **(27, 9, '2024-04-03', 169.99),**

### **(28, 11, '2024-04-05', 499.99),**

### **(29, 13, '2024-04-08', 249.99),**

### **(30, 15, '2024-04-10', 1199.99),**

### **(31, 1, '2024-04-12', 359.99),**

### **(32, 3, '2024-04-15', 799.99),**

### **(33, 5, '2024-04-18', 249.99),**

### **(34, 2, '2024-04-20', 499.99),**

### **(35, 4, '2024-04-22', 1399.99);**

### **-- Insert 15 sample records into OrderDetails table**

### **INSERT INTO OrderDetails (OrderDetailID, OrderID, ProductID, Quantity)**

### **VALUES**

### **(1, 1, 1, 2),**

### **(2, 1, 3, 1),**

### **(3, 2, 5, 1),**

### **(4, 2, 7, 1),**

### **(5, 3, 9, 3),**

### **(6, 3, 11, 2),**

### **(7, 4, 2, 1),**

### **(8, 4, 4, 1),**

### **(9, 5, 6, 1),**

### **(10, 5, 8, 2),**

### **(11, 6, 10, 1),**

### **(12, 6, 12, 1),**

### **(13, 7, 14, 1),**

### **(14, 7, 1, 1),**

### **(15, 8, 3, 2);**

### **-- Insert 15 sample records into Inventory table**

### **INSERT INTO Inventory (InventoryID, ProductID, QuantityInStock, LastStockUpdate)**

### **VALUES**

### **(1, 1, 50, '2024-02-01'),**

### **(2, 2, 30, '2024-02-01'),**

### **(3, 3, 80, '2024-02-01'),**

### **(4, 4, 20, '2024-02-01'),**

### **(5, 5, 60, '2024-02-01'),**

### **(6, 6, 40, '2024-02-01'),**

### **(7, 7, 25, '2024-02-01'),**

### **(8, 8, 70, '2024-02-01'),**

### **(9, 9, 55, '2024-02-01'),**

### **(10, 10, 45, '2024-02-01'),**

### **(11, 11, 35, '2024-02-01'),**

### **(12, 12, 65, '2024-02-01'),**

### **(13, 13, 75, '2024-02-01'),**

### **(14, 14, 15, '2024-02-01'),**

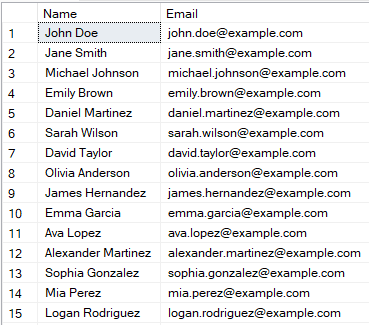
### **(15, 15, 10, '2024-02-01');**

# **Task 2:**

1. Write an SQL query to retrieve the names and emails of all customers.

**SELECT FirstName+' '+LastName as Name, Email FROM Customers;**

**Output:**

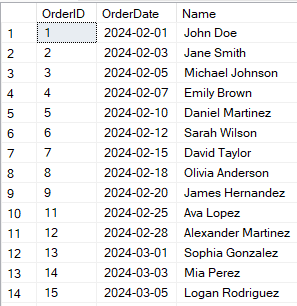
****

1. Write an SQL query to list all orders with their order dates and corresponding customer names.

**SELECT O.OrderID, O.OrderDate, C.FirstName+' '+C.LastName as Name FROM Orders as O**

**JOIN Customers AS C ON O.OrderID=C.CustomerID;**

**Output:**

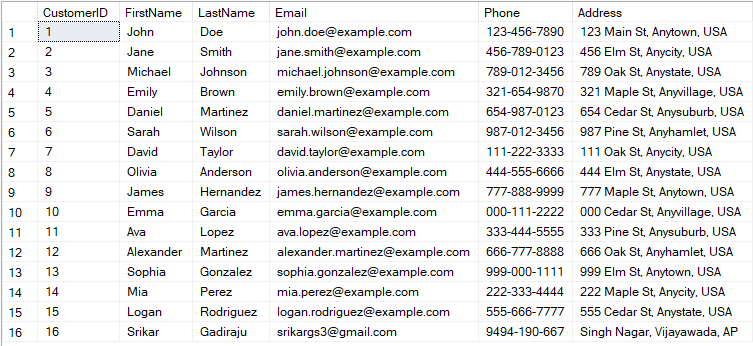
****

1. Write an SQL query to insert a new customer record into the "Customers" table. Include customer information such as name, email, and address.

**INSERT INTO Customers VALUES**

**(16, 'Srikar', 'Gadiraju', 'srikargs2@gmail.com', '9494-190-667', 'Vijayawada');**

**Output:**

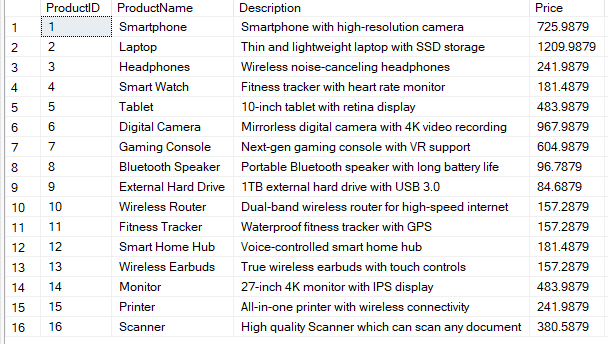
****

1. Write an SQL query to update the prices of all electronic gadgets in the "Products" table by increasing them by 10%.

**UPDATE Products**

**SET Price=Price+Price\*10/100**

**Output:**

****

1. Write an SQL query to delete a specific order and its associated order details from the "Orders" and "OrderDetails" tables. Allow users to input the order ID as a parameter.

**CREATE PROCEDURE DeleteOrderByOrderID**

**@OrderID INT**

**AS**

**BEGIN**

**DELETE FROM OrderDetails**

**WHERE OrderID = @OrderID**

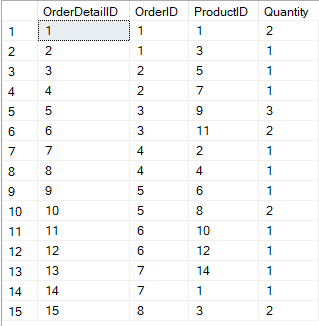
**DELETE FROM Orders**

**WHERE OrderID = @OrderID**

**END;**

**EXEC DeleteOrderByOrderID @OrderID = 15;**

**Output:**

****

1. Write an SQL query to insert a new order into the "Orders" table. Include the customer ID, order date, and any other necessary information.

**INSERT INTO Orders VALUES**

**(15, 15, '2024-03-05', 230.99000);**

**Output:**

****

1. Write an SQL query to update the contact information (e.g., email and address) of a specific customer in the "Customers" table. Allow users to input the customer ID and new contact information.

**SELECT \* FROM Customers;**

**CREATE PROCEDURE UpdateInfo**

**@CustomerID INT, @NewEmail VARCHAR(50),**

**@NewAddress VARCHAR(100)**

**AS**

**BEGIN**

**UPDATE Customers**

**SET Email = @NewEmail**

**WHERE CustomerID = @CustomerID**

**UPDATE Customers**

**SET Address = @NewAddress**

**WHERE CustomerID = @CustomerID**

**END;**

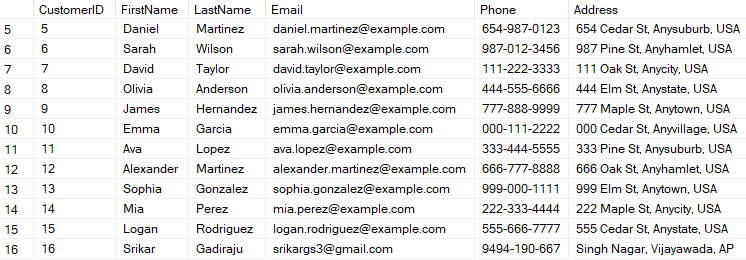
**EXEC UpdateInfo**

**@CustomerID = 16,**

**@NewEmail = 'srikargs3@gmail.com',**

**@NewAddress = 'Singh Nagar, Vijayawada, AP';**

**Output:**

****

1. Write an SQL query to recalculate and update the total cost of each order in the "Orders" table based on the prices and quantities in the "OrderDetails" table

**UPDATE Orders**

**SET TotalAmount = (**

**SELECT SUM(Quantity \* Products.Price)**

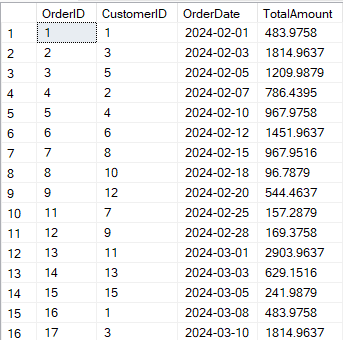
**FROM OrderDetails**

**INNER JOIN Products ON OrderDetails.ProductID = Products.ProductID**

**WHERE OrderDetails.OrderID = Orders.OrderID**

**)**

**Output:**

****

1. Write an SQL query to delete all orders and their associated order details for a specific customer from the "Orders" and "OrderDetails" tables. Allow users to input the customer ID as a parameter.

**CREATE PROCEDURE DeleteOrders**

**@CusID INT**

**AS**

**BEGIN**

**DELETE FROM Orders**

**WHERE CustomerID = @CusID**

**DELETE FROM OrderDetails**

**WHERE OrderID IN(**

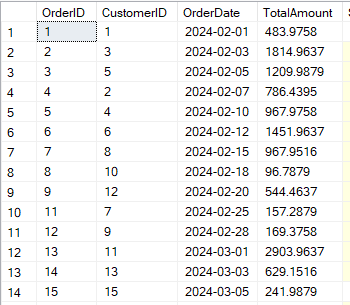
**SELECT OrderID FROM Orders**

**WHERE CustomerID = @CusID);**

**END;**

**EXEC DeleteOrders @CusID = 14;**

**Output:**

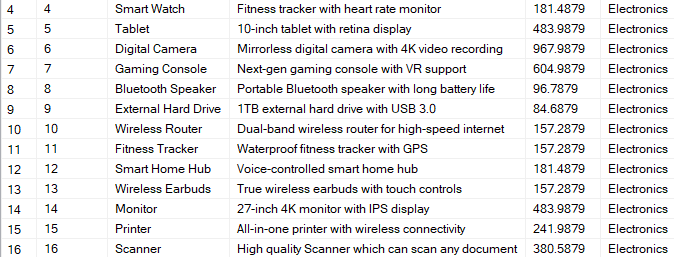
****

1. Write an SQL query to insert a new electronic gadget product into the "Products" table, including product name, category, price, and any other relevant details.

**INSERT INTO Products VALUES**

**(16, 'Scanner', 'Scans any document', 345.989);**

**Output:**

****

1. Write an SQL query to update the status of a specific order in the "Orders" table (e.g., from "Pending" to "Shipped"). Allow users to input the order ID and the new status

**ALTER TABLE Orders**

**ADD Status VARCHAR(15);**

**UPDATE Orders**

**SET Status='Pending'**

**DECLARE @OID INT = 1**

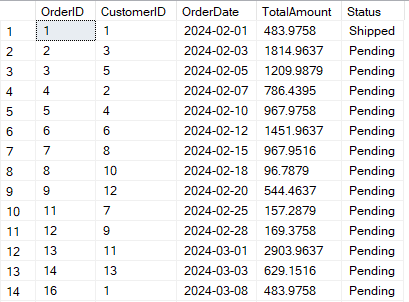
**UPDATE Orders**

**SET Status='Shipped'**

**WHERE OrderID = @OID**

**SELECT \* FROM ORDERS**

**Output:**

****

1. Write an SQL query to calculate and update the number of orders placed by each customer in the "Customers" table based on the data in the "Orders" table.

**ALTER TABLE Customers**

**ADD NoOfOrders INT**

**ALTER TABLE Customers**

**DROP COLUMN NoOfOrders**

**UPDATE Customers**

**SET NoOfOrders = COALESCE(OrdersPerCustomer.NoOfOrders, 0)**

**FROM Customers**

**JOIN (**

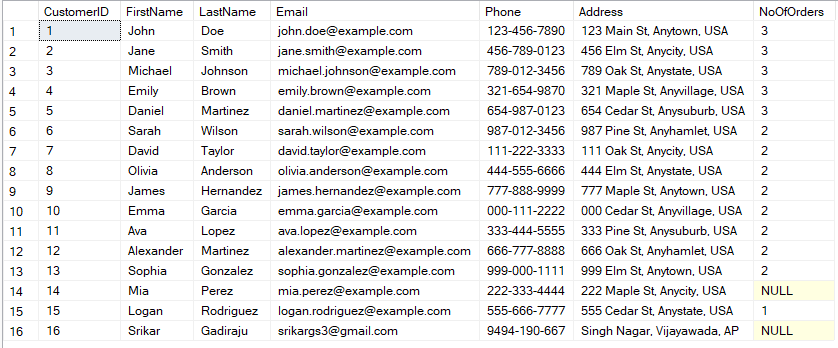
**SELECT CustomerID, COUNT(\*) AS NoOfOrders**

**FROM Orders**

**GROUP BY CustomerID**

**) AS OrdersPerCustomer ON Customers.CustomerID = OrdersPerCustomer.CustomerID;**

**Output:**

****

# **Task 3**

1. Write an SQL query to retrieve a list of all orders along with customer information (e.g., customer name) for each order.

**SELECT**

**O.OrderID,**

**O.OrderDate,**

**O.TotalAmount,**

**C.FirstName+' '+C.LastName AS CustomerName,**

**C.Email,**

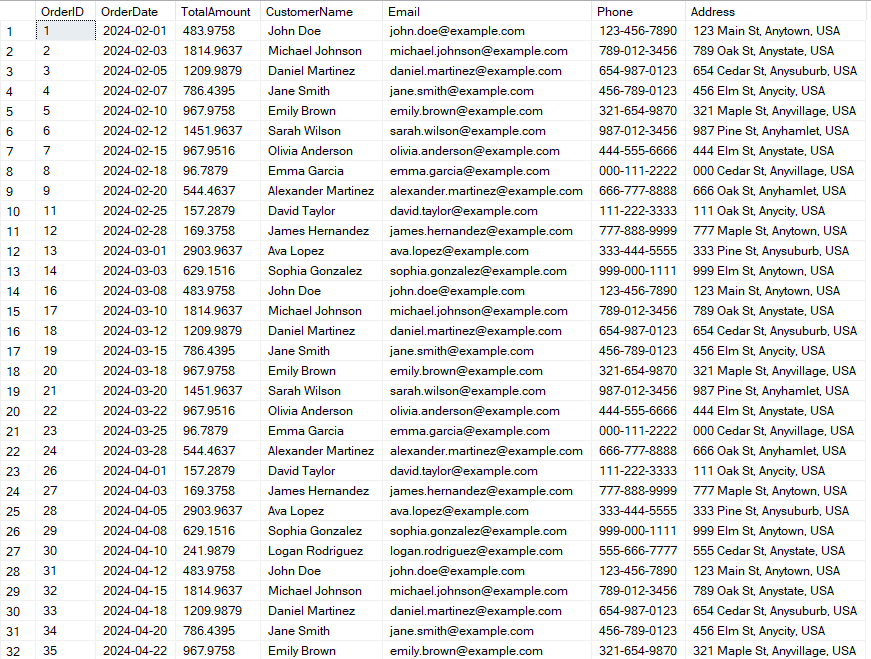
**C.Phone,**

**C.Address**

**FROM Orders AS O**

**JOIN Customers AS C ON C.CustomerID = O.CustomerID;**

**Output:**

****

1. Write an SQL query to find the total revenue generated by each electronic gadget product. Include the product name and the total revenue.

**SELECT P.ProductName,**

**SUM(OD.Quantity\*P.Price) AS Revenue**

**FROM Products AS P**

**JOIN OrderDetails AS OD ON OD.ProductID = P.ProductID**

**GROUP BY P.ProductName**

**ORDER BY Revenue DESC;**

**Output:**

****

1. Write an SQL query to list all customers who have made at least one purchase. Include their names and contact information.

**SELECT**

**C.FirstName,**

**C.LastName,**

**C.Email,**

**C.Phone,**

**C.Address**

**FROM**

**Customers AS C**

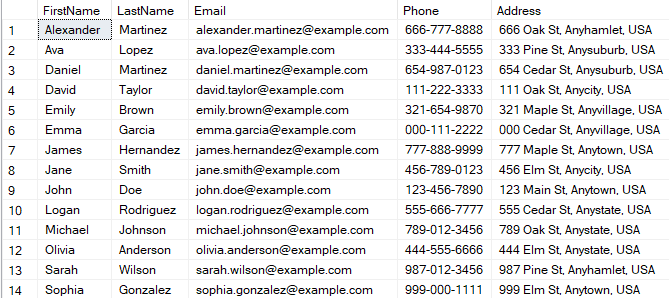
**JOIN**

**Orders AS O ON C.CustomerID = O.CustomerID**

**GROUP BY**

**C.FirstName, C.LastName, C.Email, C.Phone, C.Address**

**Output:**

****

1. Write an SQL query to find the most popular electronic gadget, which is the one with the highest total quantity ordered. Include the product name and the total quantity ordered.

**SELECT TOP 1**

**P.ProductName, OD.Quantity FROM Products AS P**

**JOIN OrderDetails AS OD**

**ON OD.ProductID = P.ProductID**

**ORDER BY OD.Quantity DESC;**

**Output:**

****

1. Write an SQL query to retrieve a list of electronic gadgets along with their corresponding categories.

**ALTER TABLE Products**

**ADD Category VARCHAR(20)**

**UPDATE Products**

**SET Category =**

**CASE**

**WHEN Description LIKE '%smartphone%' THEN 'Electronics'**

**WHEN Description LIKE '%laptop%' THEN 'Electronics'**

**WHEN Description LIKE '%headphones%' THEN 'Electronics'**

**WHEN Description LIKE '%smart watch%' THEN 'Electronics'**

**WHEN Description LIKE '%tablet%' THEN 'Electronics'**

**WHEN Description LIKE '%digital camera%' THEN 'Electronics'**

**WHEN Description LIKE '%gaming console%' THEN 'Electronics'**

**WHEN Description LIKE '%bluetooth speaker%' THEN 'Electronics'**

**WHEN Description LIKE '%external hard drive%' THEN 'Electronics'**

**WHEN Description LIKE '%wireless router%' THEN 'Electronics'**

**WHEN Description LIKE '%fitness tracker%' THEN 'Electronics'**

**WHEN Description LIKE '%smart home hub%' THEN 'Electronics'**

**WHEN Description LIKE '%wireless earbuds%' THEN 'Electronics'**

**WHEN Description LIKE '%monitor%' THEN 'Electronics'**

**WHEN Description LIKE '%printer%' THEN 'Electronics'**

**WHEN Description LIKE '%scanner%' THEN 'Electronics'**

**ELSE 'Other'**

**END;**

**SELECT ProductName, Category FROM Products**

** Output:**

1. Write an SQL query to calculate the average order value for each customer. Include the customer's name and their average order value.

**SELECT**

**C.CustomerID,**

**C.FirstName,**

**C.LastName,**

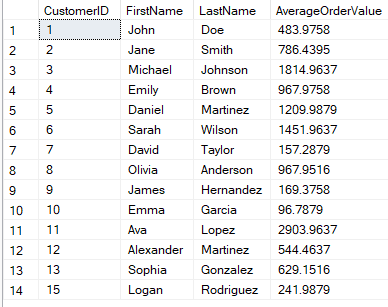
**AVG(O.TotalAmount) AS AverageOrderValue**

**FROM Customers C**

**JOIN Orders O ON C.CustomerID = O.CustomerID**

**GROUP BY C.CustomerID, C.FirstName, C.LastName;**

**Output:**

****

1. Write an SQL query to find the order with the highest total revenue. Include the order ID, customer information, and the total revenue

**SELECT TOP 1**

**C.FirstName, C.LastName,**

**O.OrderID, SUM(O.TotalAmount) AS Revenue**

**FROM Customers AS C**

**JOIN Orders AS O ON O.CustomerID = C.CustomerID**

**GROUP BY**

**C.FirstName, C.LastName, O.OrderID**

**ORDER BY Revenue DESC**  
**Output:**

****

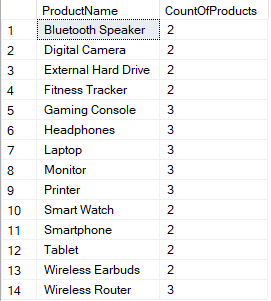
1. Write an SQL query to list electronic gadgets and the number of times each product has been ordered.

**SELECT P.ProductName, COUNT(OD.ProductID) CountOfProducts FROM Products AS P**

**JOIN OrderDetails AS OD ON OD.ProductID = P.ProductID**

**GROUP BY P.ProductName**

**Output:**

****

1. Write an SQL query to find customers who have purchased a specific electronic gadget product. Allow users to input the product name as a parameter

**CREATE PROCEDURE FindCustomer**

**@ProductName VARCHAR(20)**

**AS**

**BEGIN**

**SELECT C.FirstName+' '+C.LastName AS CustomerName,**

**P.ProductName**

**FROM Customers AS C**

**JOIN Orders AS O ON O.CustomerID = C.CustomerID**

**JOIN OrderDetails AS OD ON OD.OrderID = O.OrderID**

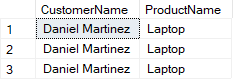
**JOIN Products AS P ON P.ProductID = OD.ProductID**

**WHERE P.ProductName = @ProductName**

**END;**

**EXEC FindCustomer @ProductName = 'Laptop'**

**Output:**

****

1. Write an SQL query to calculate the total revenue generated by all orders placed within a specific time period. Allow users to input the start and end dates as parameters.

**CREATE PROCEDURE CalcTotalRev**

**@StartDate DATE,**

**@EndDate DATE**

**AS**

**BEGIN**

**SELECT SUM(OD.Quantity \* P.Price) 'Total Revenue'**

**FROM Orders O**

**JOIN OrderDetails OD ON OD.OrderID = O.OrderID**

**JOIN Products P ON P.ProductID = OD.ProductID**

**WHERE**

**O.OrderDate>=@StartDate**

**AND O.OrderDate<=@EndDate**

**END;**

**EXEC CalcTotalRev @StartDate='2024-02-05', @EndDate='2024-02-12';**

**Output:**

****

## **Task 4:**

1. Write an SQL query to find out which customers have not placed any orders.

**SELECT**

**C.FirstName,**

**C.LastName,**

**C.Email,**

**C.Phone,**

**C.Address**

**FROM Customers AS C**

**JOIN Orders AS O ON O.CustomerID=C.CustomerID**

**WHERE O.TotalAmount=NULL**

1. Write an SQL query to find the total number of products available for sale.

**select**

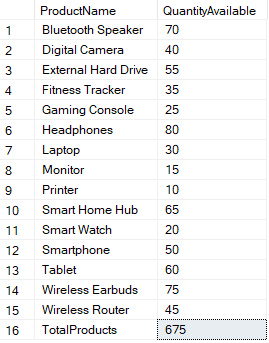
**coalesce(ProductName,'TotalProducts')as ProductName , sum(QuantityInStock) as QuantityAvailable**

**from products,Inventory**

**where products.productID=Inventory.productID**

**group by rollup (ProductName);**

**Output:**

****

1. Write an SQL query to calculate the total revenue generated by TechShop.

**SELECT SUM(TotalAmount) TotalRevenue FROM Orders;**

**Output:**

****

1. Write an SQL query to calculate the average quantity ordered for products in a specific category. Allow users to input the category name as a parameter.

**SELECT CategoryName, AverageQuantity AS OverallAverageQuantity**

**FROM (**

**SELECT P.Category AS CategoryName, AVG(OD.Quantity) AS AverageQuantity**

**FROM OrderDetails OD**

**JOIN Products P ON P.ProductID = OD.ProductID**

**WHERE P.Category = 'Electronics'**

**GROUP BY P.Category**

**) AS SubqueryResult**

**Output:**

****

1. Write an SQL query to calculate the total revenue generated by a specific customer. Allow users to input the customer ID as a parameter.

**DECLARE @CustomerID INT = 13**

**SELECT**

**CustomerID, FirstName, LastName, TotalRevenue**

**FROM(**

**SELECT C.CustomerID, C.FirstName, C.LastName, SUM(O.TotalAmount) TotalRevenue FROM Customers C**

**JOIN Orders O ON O.CustomerID = C.CustomerID**

**WHERE O.CustomerID = @CustomerID**

**GROUP BY C.CustomerID, C.FirstName, C.LastName) AS CustomerRevenue;**

**Output:**

****

1. Write an SQL query to find the customers who have placed the most orders. List their names and the number of orders they've placed

**WITH OrderCount AS(**

**SELECT**

**C.CustomerID,**

**C.FirstName,**

**C.LastName,**

**COUNT(O.OrderID) NoOfOrders**

**FROM Customers C**

**JOIN Orders O ON O.CustomerID = C.CustomerID**

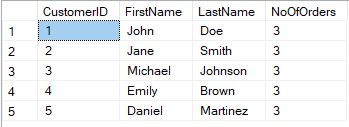
**GROUP BY C.CustomerID, C.FirstName, C.LastName**

**)**

**SELECT CustomerID, FirstName, LastName, NoOfOrders FROM OrderCount**

**WHERE NoOfOrders = (SELECT MAX(NoOfOrders) FROM OrderCount);**

**Output:**

****

1. Write an SQL query to find the most popular product category, which is the one with the highest total quantity ordered across all orders.

**SELECT TOP 1 P.ProductName, SUM(Quantity) Quantity FROM OrderDetails OD**

**JOIN Products P ON P.ProductID = OD.ProductID**

**GROUP BY P.ProductName**

**ORDER BY Quantity DESC;**

**Output:**

****

1. Write an SQL query to find the customer who has spent the most money (highest total revenue) on electronic gadgets. List their name and total spending.

**SELECT TOP 1**

**C.FirstName+' '+C.LastName CustomerName,**

**O.TotalAmount**

**FROM Customers C**

**JOIN Orders O ON O.CustomerID = C.CustomerID**

**ORDER BY O.TotalAmount DESC;**

**Output:**

****

1. Write an SQL query to calculate the average order value (total revenue divided by the number of orders) for all customers.

**WITH OrderValue AS(**

**SELECT**

**O.CustomerID,**

**C.FirstName+' '+C.LastName AS CustomerName,**

**SUM(O.TotalAmount) AS TotalRevenue,**

**COUNT(O.OrderID) AS NoOfOrders**

**FROM Orders O**

**JOIN Customers C ON C.CustomerID = O.CustomerID**

**GROUP BY O.CustomerID, C.FirstName, C.LastName)**

**SELECT**

**OV.CustomerID,**

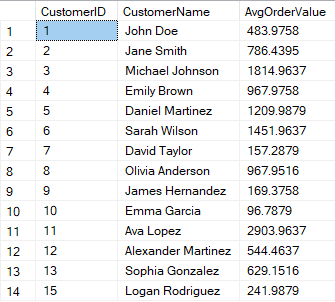
**OV.CustomerName,**

**AVG(OV.TotalRevenue/OV.NoOfOrders) AS AvgOrderValue**

**FROM OrderValue OV**

**GROUP BY OV.CustomerID, OV.CustomerName;**

**Output:**

****

1. Write an SQL query to find the total number of orders placed by each customer and list their names along with the order count.

**SELECT**

**C.FirstName,**

**COUNT(O.OrderID) TotalOrders**

**FROM Customers C**

**JOIN Orders O ON O.CustomerID = C.CustomerID**

**GROUP BY C.FirstName**

**ORDER BY TotalOrders DESC;**

**Output:**

****