SQL ASSIGNMENT

NAME: HEMASHRI S

TITLE: TECHSHOP (AN ELECTRONIC GADGETS)

```
drop database if exists TechShop;

CREATE DATABASE TechShop;
USE TechShop;
-- TASK 1
-- CUSTOMERS TABLE

CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY AUTO_INCREMENT,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Email VARCHAR(100),
    Phone VARCHAR(15),
    Address VARCHAR(255)
);

INSERT INTO Customers (FirstName, LastName, Email, Phone, Address) VALUES
('Ravi', 'Kumar', 'ravi.kumar@example.com', '9876543210', 'Chennai'),
('Priya', 'Sharma', 'priya.sharma@example.com', '8765432109', 'Mumbai'),
('Arun', 'Reddy', 'arun.reddy@example.com', '9988776655', 'Hyderabad'),
('Lakshmi', 'Devi', 'lakshmi.devi@example.com', '9871234560', 'Bangalore'),
```

('Vikram', 'Patel', 'vikram.patel@example.com', '9845123456', 'Ahmedabad'),
('Anjali', 'Mehta', 'anjali.mehta@example.com', '9823345566', 'Pune'),
('Suresh', 'Nair', 'suresh.nair@example.com', '9812345678', 'Kochi'),
('Divya', 'Verma', 'divya.verma@example.com', '9876543211', 'Delhi'),
('Rahul', 'Joshi', 'rahul.joshi@example.com', '9844112233', 'Nagpur'),
('Neha', 'Kapoor', 'neha.kapoor@example.com', '9800223344', 'Jaipur');

-- PRODUCTS TABLE

```
CREATE TABLE Products (
    ProductID INT PRIMARY KEY AUTO_INCREMENT,
    ProductName VARCHAR(100),
    Description TEXT,
    Price DECIMAL(10, 2)
);
INSERT INTO Products (ProductName, Description, Price) VALUES
('Laptop', 'Gaming laptop with high-end specs', 85000.00),
('Mouse', 'Wireless mouse', 700.00),
('Keyboard', 'Mechanical keyboard', 1500.00),
('Monitor', '24-inch Full HD monitor', 10000.00),
('Phone', 'Smartphone with AMOLED display', 30000.00),
('Charger', 'Fast USB-C charger', 1200.00),
('Tablet', '10-inch tablet', 22000.00),
```

('Webcam', 'HD webcam', 2500.00),
('Speakers', 'Bluetooth stereo speakers', 3200.00),
('Smartwatch', 'Fitness tracker smartwatch', 5000.00);

-- Orders Table

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

```
INSERT INTO Orders (CustomerID, OrderDate, TotalAmount) VALUES
(1, '2024-01-15', 85700.00),
(2, '2024-02-20', 700.00),
(3, '2024-03-05', 31500.00),
(4, '2024-01-25', 22000.00),
(5, '2024-02-18', 10000.00),
(6, '2024-02-22', 3700.00),
(7, '2024-03-10', 5000.00),
(8, '2024-01-05', 2500.00),
(9, '2024-03-15', 31200.00),
(10, '2024-03-30', 1500.00);
```

-- OrderDetails Table

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

INSERT INTO OrderDetails (OrderID, ProductID, Quantity) VALUES
(1, 1, 1),
(1, 2, 1),
(2, 2, 1),
(3, 5, 1),
(3, 3, 1),
(4, 7, 1),
(5, 4, 1),
(6, 6, 1),
(7, 10, 1),
(8, 8, 1),
(9, 5, 1),
(9, 9, 1),
(10, 3, 1);
```

-- Inventory Table

```
CREATE TABLE Inventory (
    InventoryID INT PRIMARY KEY AUTO_INCREMENT,
    ProductID INT,
    QuantityInStock INT,
    LastStockUpdate DATE,
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
INSERT INTO Inventory (ProductID, QuantityInStock, LastStockUpdate) VALUES
(1, 20, '2024-04-01'),
(2, 50, '2024-04-01'),
(3, 30, '2024-04-01'),
(4, 25, '2024-04-01'),
(5, 40, '2024-04-01'),
(6, 60, '2024-04-01'),
(7, 15, '2024-04-01'),
(8, 35, '2024-04-01'),
(9, 45, '2024-04-01'),
(10, 55, '2024-04-01');
```

-- TASK 2

-- to retrieve the names and emails of all customers.

```
SELECT FirstName, LastName, Email
```

```
FROM Customers;
```

-- 2.to list all orders with their order dates and corresponding customer names.

```
SELECT Orders.OrderID, Orders.OrderDate, Customers.FirstName, Customers.LastName
FROM Orders
JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

-- 3.to insert a new customer record into the "Customers" table

```
INSERT INTO Customers (CustomerID, FirstName, LastName, Email, Phone, Address)
VALUES (1101, 'Renu', 'vaz', 'renu.vaz@example.com', '9876993210', '123 Main St, New York, NY');
```

-- 4.to update the prices of all electronic gadgets by increasing them by 10%.

```
SET SQL_SAFE_UPDATES = 0;
UPDATE Products
SET Price = ROUND(Price * 1.10, 2);
```

-- 5. to delete a specific order and order details from the "Orders" and "OrderDetails" tables

```
DELETE FROM OrderDetails WHERE OrderID = 5001;
DELETE FROM Orders WHERE OrderID = 5001;
```

-- 6.to insert a new order into the "Orders" table.

```
INSERT INTO Orders (OrderID, CustomerID, OrderDate, TotalAmount)
VALUES (201, 1101, '2025-03-19', 250.00);
```

-- 7.to update the contact information of a specific customer in the "Customers" table.

```
UPDATE Customers
SET Email = 'new.email@example.com',
    Address = '456 New Street, Los Angeles, CA'
WHERE CustomerID = 1101;
```

-- 8.to recalculate and update the total cost of each order in the "Orders" table

```
UPDATE Orders
SET TotalAmount = (
    SELECT IFNULL(SUM(od.Quantity * p.Price), 0)
    FROM OrderDetails od
    JOIN Products p ON od.ProductID = p.ProductID
    WHERE od.OrderID = Orders.OrderID
);
```

-- 9. to delete all orders and order details for a specific customer from the "Orders" and "OrderDetails" tables

```
DELETE FROM OrderDetails
WHERE OrderID IN (
    SELECT OrderID FROM Orders WHERE CustomerID = 1101
);
DELETE FROM Orders
WHERE CustomerID = 1101;
```

-- 10.to insert a new electronic gadget product into the "Products" table, including product name, category, price, and any other relevant details.

```
INSERT INTO Products (ProductID, ProductName, Description, Price)
VALUES (201, 'Wireless Earbuds', 'Bluetooth 5.0, Noise Cancellation', 59.99);
```

-- 11. to update the status of a specific order in the "Orders" table

```
ALTER TABLE Orders ADD COLUMN Status VARCHAR(20);
UPDATE Orders
SET Status = 'Shipped'
WHERE OrderID = 1001;
```

-- 12.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 COLUMN OrderCount INT DEFAULT 0;
UPDATE Customers
SET OrderCount = (
   SELECT COUNT(*)
   FROM Orders
   WHERE Orders.CustomerID = Customers.CustomerID
);
```

-- TASK 3

-- 1.to retrieve a list of all orders along with customer information for each order.

```
SELECT
    Orders.OrderID,
    Orders.OrderDate,
    Orders.TotalAmount,
    Customers.FirstName,
    Customers.LastName,
    Customers.Email
FROM Orders
JOIN Customers ON Orders.CustomerID = Customers.CustomerID;
```

-- 2.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 TotalRevenue
FROM OrderDetails OD
JOIN Products P ON OD.ProductID = P.ProductID
GROUP BY P.ProductID, P.ProductName
ORDER BY TotalRevenue DESC;
```

-- 3.to list all customers who have made at least one purchase. Include their names and contact information.

```
SELECT
    C.CustomerID,
    C.FirstName,
    C.LastName,
```

```
    C.Email,
    C.Phone,
    C.Address
FROM Customers C
JOIN Orders O ON C.CustomerID = O.CustomerID
GROUP BY C.CustomerID, C.FirstName, C.LastName, C.Email, C.Phone, C.Address;
```

-- 4.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
    P.ProductName,
    SUM(OD.Quantity) AS TotalQuantityOrdered
FROM OrderDetails OD
JOIN Products P ON OD.ProductID = P.ProductID
GROUP BY P.ProductID, P.ProductName
ORDER BY TotalQuantityOrdered DESC
LIMIT 1;
```

-- 5.to retrieve a list of electronic gadgets along with their corresponding categories.

```
DESC Products;
ALTER TABLE Products ADD COLUMN Category VARCHAR(50);
UPDATE Products
SET Category = 'Electronics'
```

```
WHERE ProductName IN ('laptop', 'Smartwatch', 'Bluetooth Speaker','smartphone');

SELECT
   ProductName,
   Category
FROM Products
WHERE Category = 'Electronics';
```

-- 6. to calculate the average order value for each customer. Include the customer's name and their average order value.

```
SELECT
   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
ORDER BY AverageOrderValue DESC;
```

-- 7.to find the order with the highest total revenue. Include the order ID, customer information, and the total revenue.

```
SELECT
   O.OrderID,
   C.FirstName,
   C.LastName,
   C.Email,
```

```
    O.TotalAmount AS TotalRevenue
FROM Orders O
JOIN Customers C ON O.CustomerID = C.CustomerID
ORDER BY O.TotalAmount DESC
LIMIT 1;
```

-- 8.to list electronic gadgets and the number of times each product has been ordered.

```
SELECT
    P.ProductName,
    COUNT(OD.OrderID) AS TimesOrdered
FROM OrderDetails OD
JOIN Products P ON OD.ProductID = P.ProductID
WHERE P.Category = 'Electronics'
GROUP BY P.ProductID, P.ProductName
ORDER BY TimesOrdered DESC;
```

-- 9.to find customers who have purchased a specific electronic gadget product. Allow users to input the product name as a parameter.

```
SELECT
    C.CustomerID,
    C.FirstName,
    C.LastName,
    C.Email,
    C.Phone
```

```
FROM Customers C
JOIN Orders O ON C.CustomerID = O.CustomerID
JOIN OrderDetails OD ON O.OrderID = OD.OrderID
JOIN Products P ON OD.ProductID = P.ProductID
WHERE P.ProductName = 'smartwatch';
```

-- 10.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.

```
SELECT
   SUM(OD.Quantity * P.Price) AS TotalRevenue
FROM Orders O
JOIN OrderDetails OD ON O.OrderID = OD.OrderID
JOIN Products P ON OD.ProductID = P.ProductID
WHERE O.OrderDate BETWEEN '2025-03-02' AND '2025-03-07';
```

-- TASK 4

-- 1.to find out which customers have not placed any orders

```
SELECT C.CustomerID, C.FirstName, C.LastName, C.Email, C.Phone
FROM Customers C
LEFT JOIN Orders O ON C.CustomerID = O.CustomerID
WHERE O.OrderID IS NULL;
```

-- 2.to find the total number of products available for sale.

```
SELECT SUM(QuantityInStock) AS TotalAvailableProducts
FROM Inventory;
```

-- 3.to calculate the total revenue generated by TechShop.

```
SELECT SUM(OD.Quantity * P.Price) AS TotalRevenue
FROM OrderDetails OD
JOIN Products P ON OD.ProductID = P.ProductID;
```

-- 4.to calculate the average quantity ordered for products in a specific category. Allow users to input the category name as a parameter.

```
SELECT AVG(OD.Quantity) AS AverageQuantityOrdered
FROM OrderDetails OD
JOIN Products P ON OD.ProductID = P.ProductID
WHERE P.Category = 'Electronics';
```

-- 5.to calculate the total revenue generated by a specific customer. Allow users to input the customer ID as a parameter.

```
SELECT C.CustomerID, C.FirstName, C.LastName, SUM(OD.Quantity * P.Price)
AS TotalRevenue
FROM Customers C
JOIN Orders O ON C.CustomerID = O.CustomerID
JOIN OrderDetails OD ON O.OrderID = OD.OrderID
JOIN Products P ON OD.ProductID = P.ProductID
WHERE C.CustomerID = 2089
GROUP BY C.CustomerID, C.FirstName, C.LastName;
```

-- 6.to find the customers who have placed the most orders. List their names and the number of orders they've placed.

```
SELECT C.FirstName, C.LastName, COUNT(O.OrderID) AS NumberOfOrders
FROM Customers C
JOIN Orders O ON C.CustomerID = O.CustomerID
GROUP BY C.CustomerID, C.FirstName, C.LastName
ORDER BY NumberOfOrders DESC
LIMIT 10;
```

-- 7.to find the most popular product category, which is the one with the highest total quantity ordered across all orders.

```
SELECT P.Category, SUM(OD.Quantity) AS TotalQuantityOrdered
FROM Products P
JOIN OrderDetails OD ON P.ProductID = OD.ProductID
GROUP BY P.Category
ORDER BY TotalQuantityOrdered DESC
LIMIT 1;
```

-- 8.to find the customer who has spent the most money (highest total revenue) on electronic gadgets. List their name and total spending.

```
SELECT C.FirstName, C.LastName, SUM(OD.Quantity * P.Price) AS TotalSpent
FROM Customers C
JOIN Orders O ON C.CustomerID = O.CustomerID
JOIN OrderDetails OD ON O.OrderID = OD.OrderID
JOIN Products P ON OD.ProductID = P.ProductID
WHERE P.Category = 'Electronics'
```

```
GROUP BY C.CustomerID, C.FirstName, C.LastName
ORDER BY TotalSpent DESC
LIMIT 1;
```

-- 9.to calculate the average order value (total revenue divided by the number of orders) for all customers.

```
SELECT C.CustomerID, C.FirstName, C.LastName,
    SUM(O.TotalAmount) / COUNT(O.OrderID) AS AverageOrderValue
FROM Customers C
JOIN Orders O ON C.CustomerID = O.CustomerID
GROUP BY C.CustomerID, C.FirstName, C.LastName
ORDER BY AverageOrderValue DESC;
```

-- 10.to find the total number of orders placed by each customer and list their Names along with the order count.

```
SELECT
  C.CustomerID,
  C.FirstName,
  C.LastName,
  COUNT(O.OrderID) AS OrderCount
FROM Customers C
LEFT JOIN Orders O ON C.CustomerID = O.CustomerID
GROUP BY C.CustomerID, C.FirstName, C.LastName
ORDER BY OrderCount DESC;
```