Foundation training Assignment-1

Tasks 2: Select, Where, Between, AND, LIKE:

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

Code:

```
SELECT FirstName, LastName, Email FROM Customers;
```

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

Code:

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

FROM Orders O

JOIN Customers C ON O.CustomerID = C.CustomerID;
```

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

Code:

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

VALUES ('Manoj', 'Bhat', 'manoj.bhat@example.com', '9876543211', 'Coimbatore, Tamil Nadu');
```

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

Code:

```
UPDATE Products

SET Price = Price * 1.10

WHERE ProductName LIKE '%Laptop%' OR ProductName LIKE '%Smartphone%'

  OR Description LIKE '%electronic%' OR ProductName LIKE '%Tablet%';
```

5. 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.

Code:

```
DECLARE @OrderID INT = 3;

DELETE FROM OrderDetails WHERE OrderID = @OrderID;
```

```
DELETE FROM Orders WHERE OrderID = @OrderID;
```

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

Code:

```
INSERT INTO Orders (CustomerID, OrderDate, TotalAmount)
VALUES (2, GETDATE(), 12000.00);
```

7. 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.

Code:

```
UPDATE Customers
SET Email = 'lily@gmail.com',
    Address = 'Bhojpuri, MP'
WHERE CustomerID = 1;
```

8. 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.

Code:

```
UPDATE O
SET TotalAmount = (
    SELECT SUM(P.Price * OD.Quantity)
    FROM OrderDetails OD
    JOIN Products P ON OD.ProductID = P.ProductID
    WHERE OD.OrderID = O.OrderID
)
FROM Orders O;
```

9. 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.

Code:

```
DECLARE @CustomerID INT = 4;
```

```
DELETE OD
FROM OrderDetails OD
JOIN Orders O ON OD.OrderID = O.OrderID
WHERE O.CustomerID = 2;
DELETE FROM Orders WHERE CustomerID = 2;
```

10. 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.

Code:

```
INSERT INTO Products (ProductName, Description, Price)
VALUES ('Bluetooth Speaker', 'Portable electronic speaker with bass boost', 3500.00);
```

11. 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.

Code:

```
ALTER TABLE Orders ADD Status VARCHAR(50);
DECLARE @OrderID INT = 2;
DECLARE @NewStatus VARCHAR(50) = 'Shipped';
UPDATE Orders
SET Status = NULL
WHERE OrderID = 1;
```

12. 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.

Code:

```
ALTER TABLE Customers ADD OrderCount INT
UPDATE Customers
SET OrderCount = (
   SELECT COUNT(*)
   FROM Orders
   WHERE Orders.CustomerID = Customers.CustomerID
);
```