Shopify - The Online Grocery Store Embedded SQL Queries and Triggers

Name: Nishant Singh (solo project)

Roll Number: 2022328

Embedded SQL Queries:

1.) Connect to database:

```
import mysql.connector




def connect_to_database():

    try:

        conn = mysql.connector.connect(

            host="localhost",

            user="root",

            password="Password",

            database="shopify"

        )
```

```
        return conn

    except mysql.connector.Error as err:

        print("Error connecting to the database:", err)

        return None
```

2.) Place Order:

```
def place_order(customer_id, product_id, quantity, conn):

    try:

        cursor = conn.cursor()




        cursor.execute("SELECT QuantityInStock FROM Product WHERE ProductID =
%s", (product_id,))

        available_quantity = cursor.fetchone()[0]




        if available_quantity >= quantity:

            new_quantity = available_quantity - quantity

            cursor.execute("UPDATE Product SET QuantityInStock = %s WHERE
```

```
ProductID = %s", (new_quantity, product_id))




            cursor.execute("SELECT Price FROM Product WHERE ProductID = %s",
(product_id,))

            price = cursor.fetchone()[0]

            cost = price * quantity




            de_id = random.randint(1, 10)

            cursor.execute("INSERT INTO Orders (OrderDate, Status, TotalAmount,
UserID, DeliveryExecutiveID) VALUES ('2024-03-22','Processing',%s, %s, %s)",
(cost,customer_id,de_id))

            print("Order placed successfully!")

        else:

            print("Insufficient quantity in stock!")




        conn.commit()

        cursor.close()
```

```
    except mysql.connector.Error as err:


        print("Error placing order:", err)
```

3.) Inventory Analysis:

```
def inventory_analysis(conn):


    try:


        cursor = conn.cursor()


        cursor.execute("SELECT ProductID, Name, Price, QuantityInStock FROM
Product WHERE QuantityInStock < 50")


        low_stock_products = cursor.fetchall()


        print("Products with low stock(less than 50): ")


        for product in low_stock_products:


            print("ProductID:", product[0], "| Name:", product[1], "| Price:",
product[2], "| Quantity In Stock:", product[3])


        cursor.close()


    except mysql.connector.Error as err:


        print("Error performing inventory analysis:", err)
```

4.) Customer Analysis:

```
def customer_analysis(conn):


    try:


        cursor = conn.cursor()


        cursor.execute("SELECT UserID, SUM(TotalAmount) AS TotalSpent FROM
Orders GROUP BY UserID ORDER BY TotalSpent DESC LIMIT 5")


        top_customers = cursor.fetchall()


        print("Top 5 Customers based on total spent amount: ")


        for customer in top_customers:


            print("UserID:",customer[0], "| Spending:",customer[1])


        cursor.close()


    except mysql.connector.Error as err:


        print("Error performing customer analysis:", err)
```

Triggers :

Trigger 1: Alert on Low Stock
DELIMITER //

```
CREATE TRIGGER LowStockAlert
AFTER UPDATE ON Product
FOR EACH ROW
BEGIN
    IF NEW.QuantityInStock < 50 THEN
        INSERT INTO Notification (Message) VALUES (CONCAT('Low stock alert
for product: ', NEW.ProductID));
    END IF;
END;
//

DELIMITER ;
```

Trigger 2: Alert if Order is Out for Delivery

```
DELIMITER //

CREATE TRIGGER notify_user_out_for_delivery
AFTER UPDATE ON Orders
FOR EACH ROW
BEGIN
    DECLARE userName VARCHAR(255);

    -- Fetch the name of the user who placed the order
    SELECT FirstName INTO userName
    FROM User
    WHERE UserID = NEW.UserID;

    IF NEW.Status = 'Out for Delivery' THEN
        INSERT INTO Notification (Message)
        VALUES (CONCAT(userName,', Your order is out for delivery! Delivery
expected soon.'));
    END IF;
END;
//

DELIMITER ;
```