Query-1:- Reduce the price of all menu items by 10%

```
Update Menu_Item
SET Price = Price * 0.9;
```

Item_ID	Item_Name	Price	Category	Rest_ID
1	pizza	450	Italian	1
2	Burger	270	Fast Food	1
3	Sushi	320	Japanese	2
4	Pasta	360	Italian	1
5	Noodly	315	Chinese	3

Operations

Query-1:- Retrive Orders where the total is greater than 600

```
SELECT *
FROM OrdersTable;
WHERE Order_Table >600;
```

order_ID	Cust_ID	Order_Date	order_Total	Payment status
1	1	2025-01-20	800	Paid
3	3	2025-01-22	700	Paid

Result:-

The program has excuted successfully

Date:- 19/8/25

Task-4

Using Function in Queries and writing subqueries

Case study:- Online Food Ordering system

Aim:- To perform advanced query processing and best heuristics by designing optimal correlated and nested subqueries, such as finding summary statics, for the online Food Ordering system

1. Using Aggregate function with subqueries

Query1:- Find the customer(s) who placed the highest order Total

```
SELECT Cust_ID, Order_Total
FROM OrderTable
WHERE Order_Table = (select MAX(Order_Total)
  FROM orderTable);
```

Query-2:- List all the menu items whose price is above the average price of all menu items

```
SELECT Item_ID, Item_Name, Price
FROM Menu_Item
WHERE Price > (SELECT AVG(Price) FROM
    Menu_Item);
```

Output

Cust_ID	Order_Total
1	800

output

Item_ID	Item_Name	Price
3	Sushi	720

2- Nested subqueries

Query-1:- Find the names of customers who placed Orders worth more than 600

```
SELECT Cust_Name
FROM Customers
WHERE Cust_ID IN(SELECT Cust_ID FROM
Order Table WHERE Order_Total >600);
```

Query-2:- Retrives the Name of the restaurant offering the most expensive menu item

```
SELECT Rest_Name
FROM Restaurant
WHERE Rest_ID = (SELECT Rest_ID
FROM Price = (SELECT MAX(Price) FROM Menu_Item);
```

Query-1:- Retrives the category of menu item with the highest average price

```
SELECT Category
FROM Menu_Item
WHERE Category IN(
    SELECT Category
    FROM Menu_Item
    GROUP BY Category
    HAVING AVG(Price) =(SELECT MAX (Price))
        From(SELECT AVG(Price) AS AVG_PRICE
-Category FROM Menu_Item GROUP BY Category));
```

output

Cust_Name
Alice
Chastic

output

Resto_Name
Tasty Treats

output

Category
Japanese

3. Correclated Subqueries

Query-1:- Find all orders where the total is greater than the average total of all orders

```
SELECT Order_ID, Order_Total
FROM OrderTable o
WHERE Order_Total > (SELECT AVG(Order_Total)
FROM OrderTable);
```

Query-2:- Find customers who have placed more than one order

```
SELECT Cust_ID, Cust_Name
FROM Customer C
WHERE (SELECT Count(*) FROM OrderTable o
WHERE o.Cust_ID = C.Cust_ID) > 1;
```

Query-3:- Retrives the list of menuitems priced above the average price for their Category

```
SELECT Item_Name, Category, Price
FROM Menu_Item m1
WHERE Price > (
    SELECT AVG(Price)
    FROM Menu_Item m2
    WHERE m1.Category = m2.Category
);
```

Order_ID	Order_Total
1	800
3	700

Cust_ID	Cust_Name
Empty. Result if each customer has only one order	

Item_Name	category	Price
sushi	Japanese	720

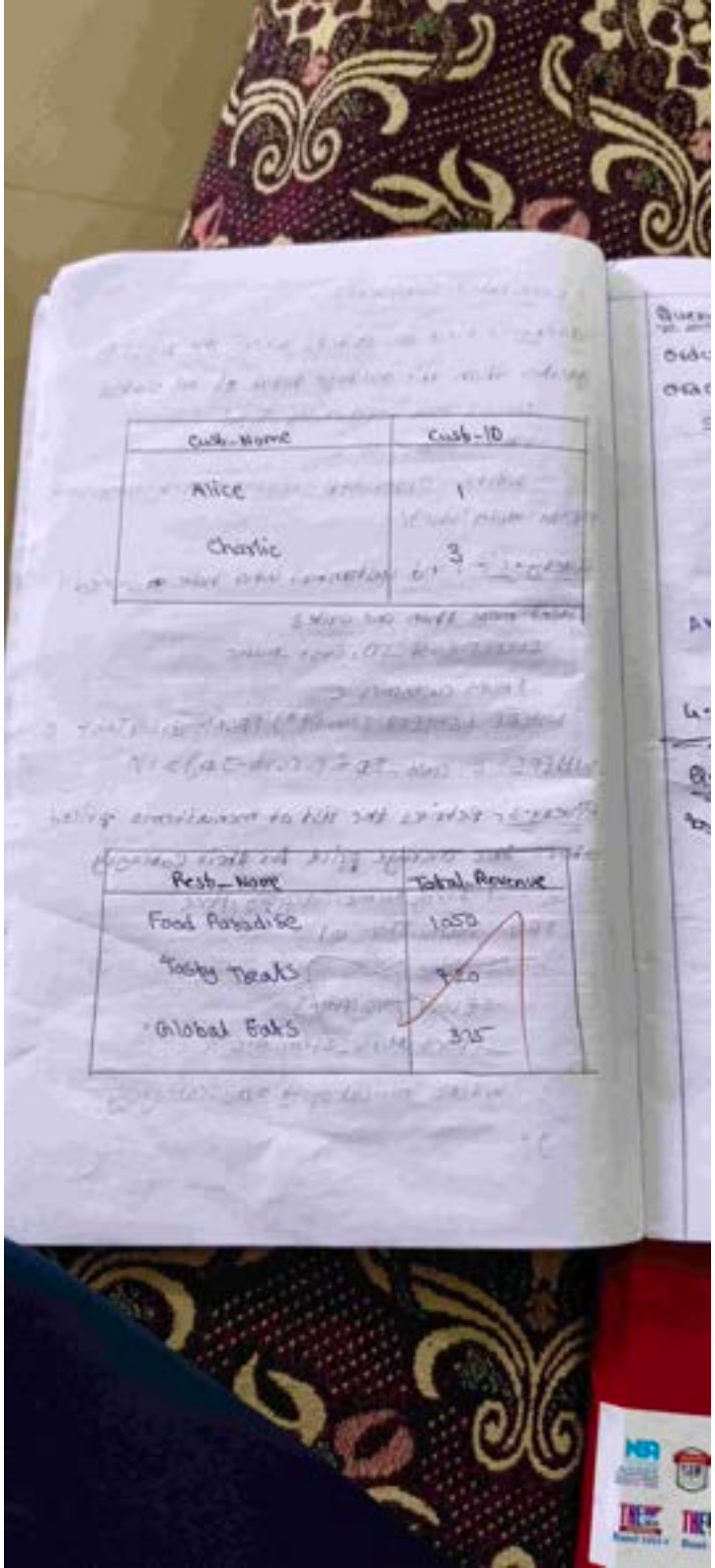
Query-3:- Find customers who have placed orders with totals higher than the average order Table of all customers

```
SELECT Cust_Name, Cust_ID
FROM Customers C
WHERE EXISTS(
    SELECT 1
    FROM OrderTable O
  WHERE C.Cust_ID = O.Cust_ID
AND Order_Total >(SELECT AVG(order_Total)
    FROM (OrderTable) );
```

4. Summary statistics with subqueries

Query-1:- Retrive the total revenue generated by each restaurant

```
SELECT Rest_Name,
(SELECT SUM(order_Total)
    FROM OrderTable O
    JOIN Menu_Item m ON O.Cust_ID = m.Rest_
    ID
WHERE m.Rest_ID = r.Rest_ID) AS Total_Revenue
FROM Restaurant r);
```



Cust_Name	Cust_ID
Alice	1
Charlie	3

Resb_Name	Total_Revenue
Food Paradise	1050
Tasty Treats	850
Global Eats	375

Query-2 :- Find the average price of menu items for each restaurant

```
SELECT Rest_Name,
    (SELECT AVG(Price)
    FROM Menu_Item m
    WHERE m.Rest_ID = r.Rest_ID) AS
Average_Price
FROM Restaurant r;
```

VEL TECH
EMP NO
PERFORMANCE (5)
RESULT ANALYSIS (5)
VIVA VOCE (5)
RECORD (5)
TOTAL (25)
VTU WITH DATE

VEL TECH - CSE
EMP NO.	4
PERFORMANCE (5)	5
RESULT ANALYSIS (5)	7
VIVA VOCE (5)	2
RECORD (5)	2
TOTAL (25)	20
VTU WITH DATE	20

Result

The program has executed successfully