Task 4 :

Using functions in Queries and writing Subqueries

Case study : Online Food Ordering System

Objective : To perform advanced query processing and test heuristics by designing optimal correlated & nested subqueries.

1. Using Aggregate functions with subqueries

Query 1 : Find the customers who placed the highest order Total

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

Query 2 : List all 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);
```

Nested Subqueries

Query 1 : Find the names of customers who placed orders worth more than 600.

```
SELECT Cust_Name
FROM customer
WHERE Cust_ID IN(SELECT Cust_ID FROM OrderTable WHERE Order_
Total >600);
```

Cust_ID	Order_total
1	800

Item_ID	Item_Name	Price
3	Sushi	720

Cust_Name
Alice
Charlie

Rest_Name
Tasty Treats

Category
Japanese

Order_ID	Order_Total
1	800
3	700

Item_Name	Category	Price
Sushi	Japanese	720

Cust_Name	Cust_id
Alice	1
Charlie	3

Query 1 : Retrieve the category of menu item with the highest average price.

```
SELECT category
FROM Menu_Item
WHERE Category IN (
   SELECT Category
   GROUP BY CATEGORY
  HAVING AVG(PRICE) = (SELECT MAX(AVG(PRICE))
        FROM(SELECT AVG(PPrice) AS AVG_PRICE, Category MENU_ITEM GROUP
                BY Category))
```

3. Correlated 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;
```

Rest_Name	Total_Revenue
Food Paradise	1050
Tasty Treats	720
Global Eats	315

Rest_Name	Average_Price
FoodParadise	360
Tasty Treats	720
GlobalEats	315

Query3 : Retrieve the list of menu items priced above the average price for their category

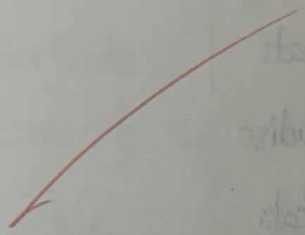
```
SELECT Item_Name, Category, Price
FROM Menu_Item m1
WHERE Price > (
 SELECT AVG (Price)
 FROM Menu_Item m2
 WHERE m1.category = m2.category
 );
```

Query4 : Find Customers who have placed orders with totals higher than the average order total of all customers.

```
SELECT Cust_Name, Cust_ID
FROM Customer (
WHERE EXISTS (
 SELECT 1
 FROM OrderTable O
WHERE m.Rest_Id = r.rest_ID) AS Total_revenue
 FROM Restaurant r;
```

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

```
SELECT r.Rest_Name,
       (SELECT AVG (PRICE))
       FROM Menu_Item m
WHERE m.Rest_Id = r.Rest_Id) As Average_price
  From Restaurant;
```



VELTECH	
EX No.	4
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	1
RECORD (3)	
TOTAL (20)	11/13
SIGN WITH DATE	

VEL TECH - CSE	
EX NO.	
PERFORMANCE (5)	
RESULT AND ANALYSIS (3)	
VIVA VOCE (3)	
RECORD (4)	
TOTAL (15)	
SIGN WITH DATE	