Resb-Name	Average_Price
Food Paradise	360
Tasby Treats	720
Global Eats	315

Task-5

Aim:- To objecti, perform advanced query processing and test if heuristics by designing optimized complex queries

1. Join Queries

Query-1:- Retrives all orders along with the corres--ponding customer's name

```
SELECT o.order_ID, o.order_Date, o.Order_Total
c.Cust_Name
FROM OrderTable.O
JOIN Customer c ON o.cust_ID = c.Cust_ID;
```

Query-2:- Retrives all Menu-items along with the restaurant name that offers them

```
SELECT m.Item_Name, m.Price, r.Rest_Name
FROM Menu_Items.m
JOIN Restaurant r ON m.Rest_ID = r.Rest_ID;
```

Query-3:- Retrives all order and their delivery status

```
SELECT o.order_ID, o.Order_Total, d.Delivery_
Status, d.Delivery_Time
FROM OrderTable.o
LEFT JOIN Delivery d ON o.order_ID =
d.Order_ID;
```

Order_ID	Order_Date	Order_Total	Cust_Name
1	2025-01-20	800	Alice
2	2025-01-21	500	Bob
3	2025-01-22	700	charlie

Item_Name	Price	Rest_Name
Pizza	450	Food Paradise
Burger	270	Food Paradise
sushi	720	Tasty Treats
Pasta	360	Food Paradise
Noodles	315	Global Eats

Order_ID	Order_Date	Delivery_Status	Delivery_Time
1	800	Delivered	2025-01-20 14:30:00
2	500	Pending	NULL
3	700	Delivered	2025-01-22 16:00:00

INNER JOIN

An Inner Join retrives records that have matching values in both tables

Query:- Retrieves all orders along with their customer names

```
SELECT o.Order_ID, o.Order_Date, o.order_Total,
c.Cust_Name
FROM OrderTable o
INNER JOIN Customer c ON o.Cust_ID=c.Cust_ID;
```

LEFT Outer JOIN

An left outer Join retrives all records from the left table and the mached records from the right table, if no match is found, NULL is returned for colowns from R.T

Query:- Retrives all customers, even those who haven't Placed any Orders

```
SELECT c.Cust_Name, o.Order_ID, o.order_Total
FROM Customers c
LEFT JOIN orderTable o ON c.Cust_ID=o.Cust_ID;
```

Right Outer JOIN

it retrives the records from right table matched records from left Table, if not it gives NULL

Query:- Retrives all orders and name of cust who placed them Include orders even, if the customer details are missing

```
SELECT o.Order_ID, o.Order_Total, c.Cust_Name
FROM OrderTable o
RIGHT JOIN Customer c ON o.Cust_ID = c.Cust_ID;
```

order_ID	order_Date	order_Total	Cust_Name
1	2025-01-20	800	Alice
2	2025-01-21	500	Bob
3	2025-01-22	700	charlie

Cust_Name	Order_ID	Order_Total
Alice	1	800
Bob	2	500
charlie	3	700

order_ID	order_Total	Cust_Name
1	800	Alice
2	500	Bob
3	700	charlie

Full Outer Join

it retrives all records from both tables if no match is found NULL is returned

Query:- Retrives all customers and all orders, even if there is no match

```
SELECT c.Cust_Name, o.Order_ID, o.Order_Total
FROM Customers c
FULL OUTER JOIN OrderTable o ON c.Cust_ID = o.Cu
```

CROSS JOIN

it retrives the cartesian product of two table.

Query:- Retrives all possible combinations of customers and menuitems

```
SELECT c.Cust_Name, m.Item_Name, m.Price
FROM Customers c
CROSS JOIN Menu_Item m;
```

SELF JOIN

A SELF JOIN joins a table with itself it is useful for hierarchical (or) comparision data.

Query:- Retrive all menu items that belong to the same restaurant as another item

```
SELECT m1.Item_Name AS Item1, m2.ItemName A
Item 2, r.Rest_Name
FROM Menu_Item m1
JOIN Menu_Item m2 ON m1.Rest_ID = m2.Rest_ID AND
m1.Item_ID != m2.Item_ID
JOIN Restaurant r ON m1.Rest_ID = r.Rest_ID;
```

Cust_Name	Order_ID	Order_Total
Alice	1	800
Bob	2	500
Charlie	3	700

Cust_Name	Item_Name	price
Alice	Pizza	450
Alice	Burger	270
Alice	Sushi	720
Alice	Pasta	360
Alice	Noodles	315
Bob	Pizza	450
Bob	Burger	270
Bob	Sushi	720
Bob	Pasta	360
Bob	Noodles	315

Item_1	Item_2	Rest_Name
Pizza	Burger	Food Paradise
Pizza	Pasta	Food paradise
Burger	Pizza	Food paradise
Burger	Pasta	Food Paradise
Pasta	Pizza	Food Paradise
Pasta	Burger	Food Paradise

2. Equivalent Queries

Query-1':- Retrives all customers who placed orders using a join (equivalent to a subquery)

using join

```
SELECT DISTINCT c.cust_Name
FROM Customer c
JOIN OrderTable o ON c.Cust_ID = o.cust_ID;
```

Equivalent subquery;

```
SELECT Cust_Name
FROM Customer
WHERE Cust_ID IN (SELECT cust_ID FROM OrderTable);
```

3. Recursive Queries

Query-1':- Generate a recursive query to find all ancestors of a given category in a hypothetical "Menu category" table.

```
CREATE TABLE Menu_Category (
    Cat_ID INT PRIMARY KEY,
    Cat_Name VARCHAR(30),
    Parent_Cat_ID INT
);
INSERT INTO Menu_Category (Cat_ID, Cat_Name, Parent_Cat_ID) VALUES (1, 'Food', NULL);
INSERT INTO Menu_Category (Cat_ID, Cat_name, Parent_Cat_ID) VALUES (2, 'Italian', 1);
```

Cust-ID	Cust_Name	Parent_Cust_ID
4	Pizza	2
2	Italian	1
1	Food	NULL

Recursive Query

```
with category Hierachy ASC
SELECT cat_ID , cat_Name , parent_cat_ID
FROM Menu_category
WHERE cat_Name = 'pizza'
UNION ALL
  SELECT mc.cat_ID , mc.parent_cat_ID
  FROM Menu_Category mc
  JOIN Category Hierachy ch ON mc.cat_ID
  =ch.parent_cat_ID)
  SELECT *FROM category Hierachy;
```

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