Date:- 8/9/25

Task 5:- Writing Join queries, equivalent AND/OR Recursive queries.

Aim:- To implement and execute IN queries equivalent queries and recursive queries.

Procedure:-

1. Create a database and table.
2. Insert sample data.
- Write equivalent queries (different approach to get the same result).
- Write SQL queries using different types of JOIN.
- Implement a recursive query (using with Recursive).
- Display resulting and verify correctness.

Different types of SQL JOINS

- (INNER) JOIN.

Tables SELECT column_name(s) from table1 INNER JOIN table2 ON table1.column_name = table2.column_name;

- LEFT (OUTER) JOIN:-

SELECT column_name(s) from table1 LEFT JOIN table2 ON table1.column_name = table1.column_name;

- RIGHT (OUTER) JOIN:-

SELECT column_names(c) from table1 RIGHT JOIN table2 on table1.columnname = table2.column_name;

• FULL (OUTER) JOIN:-

FULL OUTER JOIN table2 ON table1.column-name = table2.column-name;

1. JOIN Queries (All types)

```
CREATE TABLE Departments(
    Dept ID INT Primary KEY,
    Dept Name VARCHAR(50)
);
CREATE TABLE Bills(
    Bill ID INT PRIMARY KEY,
    Pat ID INT,
    Bill Amount DECIMAL(10,2)
    FOREIGN KEY (Referring Pat ID)
    REFERCES Patient (Pat ID)
);
```

2. Department:

Dept ID	Dept name

Patients

Pat ID	Pat Name	Dept ID

Table Bills

Bill ID	Pat ID	Bill Amount

Referrals

Referral ID	Referring Pat ID	Referred Pat ID.

2. INSERT SAMPLE DATA

```
INSERT INTO Departments VALUES
(101, 'cardiology', '102', 'oncology'),
(103, 'Pediatrics');
```

DeptID	DeptName
101	cardiology
102	oncology
103	Pediatrics

```
INSERT INTO Patients VALUES
(1, 'Alice', 101),
(2, 'Bob', 102),
(3, 'Charlie', 101),
(4, 'David', 103),
(5, 'Emma', 104);
```

PatID	PatName	DeptID
1	Alice	101
2	Bob	102
3	Charlie	101
4	David	103
5	Emma	104

```
INSERT INTO Bills VALUES
(1, 1, 5000.00);
(2, 1, 300.00)
(3, 2, 7500.00);
(4, 3, 6000.00);
```

Bill ID	Pat ID	Bill Amount
1	1	5000.00
2	1	300.00
3	2	7500.00
4	3	6000.00

INSERT INTO Referrals values
(1,1,3),
(2,3,4);

ReferralID	ReferringPatID	ReferredPatID
1	1	3
		4
2	3	

3. JOIN Queries (ALL TYPES)

a) INNER JOIN

query:-

```
SELECT p.PatName, d.DeptName
FROM Patients p.
INNER JOIN Departments
d ON p.Dept.ID = d.DeptID;
```

output:-

PatName	Dept Name
Alice	cardiology
Bob	oncology
Charlie	cardiology
David	Pediatrics

b) LEFT JOIN

query

```
SELECT p.PatName, d.DeptName from
Patients p
LEFT JOIN Departments d ON p.DeptID =
d.DeptID;
```

output:-

Pat Name	Dept Name
Alice	cardiology
Bob	oncology
Charlie	cardiology
David	Pediatrics
Emma	NULL

c, Right JOIN

Query:-

SELECT P.Pat Name, d.Dept Name FROM Patients P.

Right JOIN Departments d ON p.Dept ID = d.Dept ID;

output:-

Pat Name	Dept Name
Alice	cardiology
Bob	oncology
Charlie	cardiology
David	Pediatrics.

d, FULL OUTER JOIN

query:-

SELECT P.patName, d.Dept Name
FROM Patients P
FULL OUTER JOIN Departments d ON
P.Dept ID = d.Dept ID;

output:-

PatName	DeptName
Alice	cardiology
Bob	oncology
charlie	cardiology
David	Pediatrics
Emma	NULL

c, CROSS TO IN

Query:-

```
SELECT p.PatName ,d DeptName
FROM Patients P
CROSS JOIN Departments d;
```

output:-

patName	DeptName
Alice	cardiology
Alice	oncology
Alice	Pediatrics
Bob	cardiology
Bob	oncology
Bob	Pediatrics.
Bob	cardiology
charlie	oncology
charlie	Pediatrics.
charlie	cardiology
David	oncology
David	Pediatrics
David	cardiology
Emma	oncology
Emma	Pediatrics
Emma	

f, Self JOIN:-

Query :-

```
SELECT P1.PatName AS patient1, P2.
```

Pa.Name As Patients 2, P1.DeptID
from .patients -P1
JOIN Patient P2 ON P1.Dept ID= P2.Dept ID
WHERE P1.Pat ID < P2 .PatID;
output:-

Patient1	Patients2	DeptID

4. Equivalent queries.-
JOIN vs subquery
Using Joins
SELECT p.PatName , d. Dept Name
FROM Patients P
JOIN Departments don P.Dept ID= doDept ID;
using a subquery
SELECT PatName,
(SELECT Dept Name FROM Department
d.Dept ID= PDept ID) As DeptName
From patients P;
output:-

Pat Name	Dept Name
Alice	cardiology
Bob	oncology
Charlie	cardiology
David	Pediatrics.

5-Recursive Query
Query:-
WITH RECURSIVE Referral chain As (
SELECT .ReferringPatID , ReferredPat
FROM Referrals

UNION

```
SELECT r.ReferringPatID, r.ReferredID,
FROM Referrals r
JOIN ReferralChain r.ReferredPatID = rc.ReferringPatID
```

)

```
SELECT rc.ReferringPatID AS InitialReferred,
       rc.ReferredPatID AS ReferredPatient,
       p1.PatName AS InitialReferredName,
       p2.PatName AS ReferredPatientName
FROM ReferralChain rc
JOIN Patients p1 ON rc.ReferringPatID = p1.PatID
JOIN Patients p2 ON rc.ReferredPatID = p2.PatID;
```

InitialReferrer	ReferredPatient	InitialReferredName	ReferredPatientName
1	3	Alice	Charlie
3	4	Charlie	David
1	4	Alice	David

VEL TECH-CSE	
EX NO	05
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	4
RECORD (5)	-
TOTAL (20)	14
SIGN WITH DATE	

8/19/25

Result: Thus, the implementation of JOIN queries, equivalent queries and recursive queries are executed successfully.

UNION

```
SELECT r.ReferringPatID, r.ReferredID,
FROM Referrals r
JOIN ReferralChain rc ON r.ReferredPatID = rc.ReferringPatID
);
SELECT rc.ReferringPatID AS InitialReferred,
       rc.ReferredPatID AS ReferredPatient,
       p1.PatName AS InitialReferedName,
       p2.PatName AS ReffferedPatientName
FROM ReferralChain rc
JOIN Patients p1 ON rc.ReferringPatID = p1.PatID
JOIN Patients p2 ON rc.ReferredPatID = p2.PatID;
```

Initial refer	referred patient	Initial referred name	Reffered patient name.
1	3	Alice	Charlie
3	4	Charlie	David
1	4	Alice	David.

VEL TECH-CSE	
EX NO	05
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	4
RECORD (5)	-
TOTAL (20)	14
SIGN WITH DATE	

8/9/25

Result:- Thus, the implementation of JOIN queries, equivalent queries and recursive queries are executed successfully.