Date: 8/9/2025

Task: Writing Join queries, Equivalent AND/OR RECURSIVE Queries

Aim: To implement and execute JOIN queries equivalent queries and recursive queries

Procedure:

1. Create a database and table
2. Insert sample data
3. Write SQL queries using different types of JOIN
4. Write equivalent queries (different approach to get the same result)
5. Implement a recursive query (using WITH RECURSIVE)
6. Display results 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 = table2 column_name;
- RIGHT (OUTER) JOIN:
  SELECT column_name(s) FROM table1

RIGHT JOIN table2 ON table1.column_name
= table2.column_name;

- FULL (OUTER) JOIN:

FULL OUTER JOIN table2 ON table1
table1.column_name = table2.column_name;

1. JOIN Queries (All Types)

CREATE TABLE departments(
DeptID INT PRIMARY KEY,
deptname VARCHAR(50)
);

CREATE TABLE Bills(
BillID INT PRIMARY KEY,
PatID INT,
BillAmount DECIMAL(10,2)
FOREIGN KEY (ReferringPatID)
REFFERCES Patient(PatID)
);

2 departments:

DeptID	DeptName

Patients

PatID	PatName	Dept ID

Table Bills

BillID	PatID	BillAmount

Referrals

ReferalID	ReferringPatID	ReferredPatID

2. INSERT SAMPLE DATA

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

DepID	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	DEPID
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, 3000.00),
(3, 2, 7500.00);
(4, 3, 6000.00);

BillID	PatID	BillAmount
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	Reffered PatID
1	1	3
2	3	4

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:

Pat Name	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.PatName, d.DeptName
FROM Patients P
RIGHT JOIN Departments d ON
    P.DeptID=d.DeptID;
```

Output:

PatName	DeptName
Alice	cardiology
Bob	oncology
Charlie	cardiology
David	Pediatrics

d, FULL OUTER JOIN

Query:

```
SELECT  P.PatName, d.DeptName
 FROM   Patients P
FULL OUTER JOIN Departments d ON
                              P.DeptID = d.DeptID;
```

OUTPUT:

PatName	DeptName
Alice	cardiology
Bob	oncology
Charlie	Cardiology
David	Pediatrics
Emma	NULL

e, CROSS JOIN

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
charlie	cardiology
charlie	oncology
charlie	Pediatrics
David	cardiology
David	oncology
David	Pediatrics
Emma	cardiology
Emma	oncology
Emma	Pediatrics

f, Self JOIN:

Query:

```
SELECT P1.PatName AS Patient1, P2.PatName AS
                                Patient2, P1.DeptID
FROM Patients P1
JOIN Patient P2 ON P1.DeptID=P2.DeptID
WHERE P1.PatID < P2.PatID;
```

OUTPUT:

Patient 1	Patient 2	Dept ID

4. Equivalent queries:

JOIN vs. subquery

Using JOINS

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

Using a subquery

```
SELECT PatName,
    (SELECT DeptName FROM Department
     d.DeptID = p.DeptID) AS DeptName
FROM Patients p;
```

Outputs

PatName	DeptName
Alice	cardiology
Bob	oncology
Charlie	cardiology
David	pediatrics

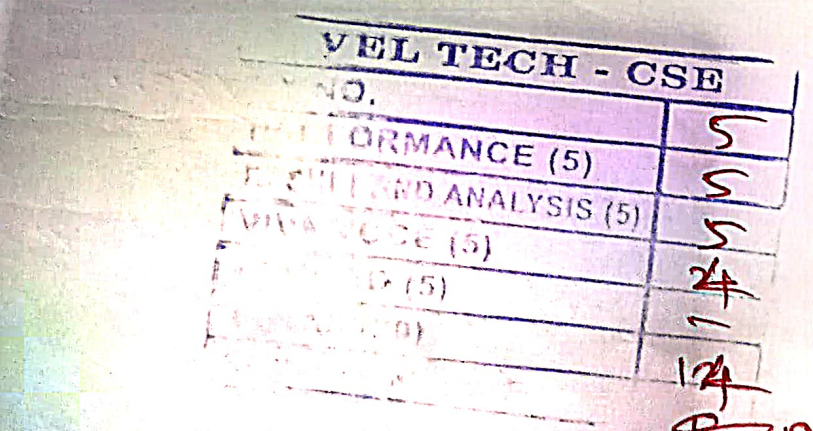
5. Recursive query

Query:

```
WITH RECURSIVE ReferralChain AS (
    SELECT ReferringPatID, ReferredPat
    FROM Referrals
    UNION
    SELECT rc.ReferringPatID, r.ReferredPatID
    FROM Referrals r
    JOIN ReferralChain rc ON r.ReferredPatID
        = rc.ReferringPat
)
```

```
SELECT RC.ReferringPatID AS InitialReferral,
       RC.ReferredPatID AS ReferredPatient,
       P1.PatName AS InitialReferralName,
       P2.PatName AS ReferredPatientName
FROM ReferralChain RC
JOIN Patients P1 ON RC.ReferringPatID = P1.PatID
JOIN Patients P2 ON RC.ReferredPatID = P2.PatID;
```

InitialReferral	ReferredPatient	InitialReferralName	ReferredPatientName
1	3	Alice	Charlie
3	4	Charlie	David
1	4	Alice	David



8/9/25

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

8/9