Task 5: WRITING JOIN QUERIES, EQUIPMENT AND/OR) RECURSI VE QUERIES 16/09/25

Aim:

To implement and Execute join queires, Equivalent queries and recursive queries using a universily database Scenerio

Procedure:

The SQL Joins clause is used to Combine records from two or more table in a database. A join is a means for Combing fields from two tables by using values Common values.

- * Create the database and tables students, Department, Courses Enrolment).
- * Write SQL Queries using different types of Join
- * Insert sample data.
- * Display results and verify Corectness.

Syntax:

- Select column, Column 2, column 3, from table name, table name 2 where name, column = table name 2 column Name;

Types of joins

1) Simple joins
2) Self join
3) outer join

1) inner join: returns records that have matching values in both tables SELECT column name(s) from table1 inner join table 2 on table1.column from the left table and

(Left OUTER) Join:

Return all reserve from the left and the records from the guide.

```
SELECT m.phone.id., m.brand, m.model, s.rom,
s. storage, s. battery.
FROM mobile phones m
LEFT JOIN phone specifications on m.phone.
        id = s.phone = id
```

```
INSERT INTO payment values (30000, 15000, 05000,
        2025-06-19)
```

1) JOIN QUERIES

a) INNER JOIN

```
SELECT m.phone.id, m.brand, m.model,
s.rom. storage. s.battery.
FROM. mobile phone
inner join phone specification on m.phone.id =
s.phoneid
```

b) LEFT JOIN

```
SELECT m.phone.id.m brand, m.model,
s.rom, s-storage. s-battay.
Left join phone Specification on m.phone.
id = s.phone id;
```

c) Right ~~phone~~ Join

```
SELECT m phone id, m.brand, m.model,
s.ram, s storage, s battery
from mobile phones m
Right Join.
```

Phone id	brand	model	price	roon	storage	batte
1	Realme	14pro	30000	16GB	256GB	3000
2	Redmi	10pro	15,000	8GB	128GB	4500
3	vivo	T3pro	25,000	12GB	256GB	5500

RIGHT (OUTER JOIN)

Phone id	brand	model	price	ram	storage	battery
1	realme	14pro	30000	16GB	256GB	5000mh
2	redmi	10pro	15000	8GB	128GB	4500ma
3	vivo	r3pro	25000	12GB	256GB	5500m

FULL OUTER JOIN :-

Phone id	brand	model	price	ram	storage	battery
1	realme	14Pro	30,000	16GB	256GB	5000mAh
2	redmi	10pro	15,000	8GB	128GB	2,500mAh
3	vivo	T3Pro	25,000	12GB	256GB	5500mAh

9. JOIN QUERIES

CREATE TABLES

```
Create Tables customer (
 Cust ID INT PRIMARY KEY,
 CUST NAME VARCHAR (50) NOT NULL;
 );
 Create table mobile (
  mobile ID INT PRIMARY KEY;
  Brand VARCHAR (50) Not Null;
  Mobile VARCHAR (50) Not Null;
  price.Decimal (10,2) CHECK.(prize.3000);
  );
 CREATE TABLE Purchase (
 purchase ID INT PRIMARY KEY
 cust ID INT PRIMARY KEY
 Mobile ID NOT NULL;
 Quantity INT CHECK.(Quantity 20);
 Purchase Date.Date Default.Curent.DATE
 );
 CREATE TABLE PAYMENT (
 PAyment INT PRIMARY KEY,
 PURCHASE ID INT UNIQUE,
 Payment Date Default,
 Current data
```

2) (Right outer) Join):

```
SELECT m-phone-id, m.brand- m.model- s ram.
 s.storage, s.battery
 FROM mobile phones m
 FULL.OUTER JOIN Phone specification
 on m.phone-id- s.phone-id;
```

3) FULL OUTER JOIN :

```
SELECT; m.phone ,m-brand ,m-model, s-ram,
    s: storage - s.battAy
 FROM  mobile phones m
 FULL OUTER JOIN ·phone specification on m.
 phone -id · s. phone - id ;
```

~~phone-id~~	brand	~~model~~	~~price~~	~~ram~~	~~storage~~	~~battery~~
⊢	realme	14 pro	30			

4) Equivalent Queries

```
SELECT s.mobile Name M. Model Name
 FROM · Mobile phone
 JOIN · Brand ON sphone ID = M. phone ID·
      using subquery
  SELECT  Mobile Name;
    (SELECT · Brand Name from ·B rand B·
    where M. phone ID = S.phone ID) AS
     Model Name from · Mobile phone;
```

5) Recursive Query (purchase hierachhy)

```
with Recursive purchases.
SELECT ~~PAY~~ Payment ID, Phone ID
From · pre requesties
UNION
SeLECT, Payment ID. cphone ID
FROM Prereques p
```

V EL	5
NO.	5
RFOR (5)	5
SULT AND SIS (5)	5
VIVA VOCE (5)	-
ECORD (5)	15
TOTAL (20)	
SIGN WITH E	1619

1619

Result:- The implementiori of SQL Commands using Joins and recursive queries are Executed Sucessfully