Task 5: WRITING QUERIES EQUIVALENT : AND/OF RECURSIVE QUERIES

Aim: To implement and execute Join queries equivalent queries and recursive queries using Mobile database

INNER JOIN:

Return records that matching values in both tables
SELECT m.phone_id, m.brand, m.model, s.ram,
s.storage, s.battery
FROM Mobilephones m
INNER JOIN phone spea

phone_id	brand	model	price
1	Realme	14 pro	30,000
2	Redmi	10 pro	15,000
3	vivo	T3 pro	25,000

INNER JOIN phone specifications.
ON. M. phone_id = s.phone_id;

Phone_id	Ram	storage	battery
1	16GB	256GB	5000 mAh
2	8GB	128GB	4000 mAh
3	12 GB	256GB	5500 mAh

LEFT (Outer) Join: Return all records from the table and the matched records from the right table
SELECT m.phone_id, m.brand, m.model, s.ram
s.storage, s.battery.
FROM. Mobile phones
LEFT JOIN. phone.specifications s. ON. M. Phone_id
= s. phone_id

Phone_id	brand	model	price
1	realme	14 pro	30,000
2	redmi	10 pro	15,000
3	vivo	T3 pro	25,000

ram	storage	battery
16GB	256GB	5000mAh
8GB	128GB	4500mAh
12GB	256GB	5500mAh

RIGHT (OUTER) JOIN: Return all records from the right-table and the matched records from the left-table

SELECT M.phone_id, m.brand, m. model, S.ram, S. storage, S.battery

FROM Mobilephones m
RIGHT JOIN Phone specifications
ON M.phone_id = S.phone_id;

Phone_id	brand	model	price	ram	storage	battery
1	realme	14Pro	30,000	16GB	256GB	5000mAh
2	Redmi	10pro	15,000	8GB	128GB	4500mAh
3	vivo	T3pro	25,000	12GB	256GB	5500 mAh

FULL (OUTER) JOIN: Return all records when there is a match in either left to Right table

SELECT: m.phone_id, m.brand, m.Model, S.ram, S.Storage, S.battery

FROM Mobilephones M
FULL OUTER JOIN phone specifications. S ON
m_phone_id = S.phone_id;

Phoneid	brand	model	price	ram	storage	battery
1	realme	14Pro	30,000	16GB	256GB	5000
2	Redmi	10Pro	15,000	8GB	128GB	4500
3	vivo	T3pro	25,000	12GB	256GB	5500

U JOIN Queries

CREATE TABLES

```
Create Tables Customer(
  Cust ID IN PRIMARY KEY;-
  Cust Name  VARCHAR(50) NOTNULL;
  );
  Create table Mobile(
    Mobile ID INT PRImary KEY;
    Brand VARCHAR(50) NOTNULL;
    Modial VARCHAR(50) NOTNLL;
   price DECIMAL(10,2) CheckPrice= 30,000);
);
Create Table purchase(
    purchase ID INT PRIMary KEY;
     CUST ID NOT NULL;
     MobileID NOT NULL;
  Quantity INT CHECK (Quantity >0);
Purchase Date DATE DEFAULT CURRENTDATE;
   FORGIN KEY (Cust ID);
   REFERENCES Mobiles (MobileID)
);
Create Table payment(
     payment ID INT PRIMARY KEY;
     purchase ID INT UNIQUE;
     Amount DECIMAL(10,2) NOTNULL;
     Payment Date DATE DEFAULT;
      CURRENT_DATE;
Payment method VARCHAR(20)
CHECK (paymentmethod INT(UPI|ard;
         Net banking; (100);
      FORIEGN KEY (purchase id
      REFERENCES purchase(purchase id)
  );
```

2. INSERT SAMPLE DATA

Insert into Mobile value Android items);

(101, 'Realme');

(102, 'Redmi');

(103, 'vivo');

insert INTO Mobile value 'Payment values

(1, 'Realme'; 101);

(2, 'Redmi'; 102);

(3, 'vivo'; 101);

(4, 'poco'; 103);

(5, 'iqoo' 104); -- Invalid phoneID for

OUTER JOIN example

INSERT INTO Re, values

('c1', 'Database System', 101);

('c2', 'Good product & worth it; 101);

('c3', 'product its good', 102);

('c4', 'afford to buy it 103);

INSERT INTO Payme value (30,000, 15000, 2500, 2025-08-19)

1 Row created completed.);

Result : Record inserted Successfully

3. JOIN QUERIES:

a) INNER JOIN

SELECT m.phone_id, m.brand, m.model, s.ram, s.storage, s.battery

FROM Mobilephone m

INNER JOIN phone specification ON M. phone_id: s.phone_id:

b) LEFT JOIN

```
SELECT m.phone_id, m.brand, m.model, s.ram, storage,
 s.battery.
FROM Mobile phones.m
 LEFT JOIN phone specification ON M.phone_id =
           s.phone_id;
```

c) RIGHT JOIN

```
SELECT m.phone_id, m.brand, m.model, s-ram, s-storage,
    s.battery
 from mobile phones m
  RIGHT JOIN Phone specification
    ON m.phone_id = s.phone_id;
```

d) FULL 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!
```

4) Equivalent Queries

```
SELECTS: Mobile Name, Model Name.
 from Mobile phone
JOIN Brand ON phoneID - m-phoneID;
```

Using subquery

```
SELECT MobileName
  (select BrandName FROM Brand B
  WHRE M.PhoneID = S.PhoneID) AS ModelName
 FROM Mobile Phone;
```

5) RECOURSIVE QUERY

```
WITH RECURSIVE purchasesAC
SELECT PaymentID,      ID
From prereqursites     phone
UNION
                       phone
SELECT PaymentID,      C.ID
From prerequireites
JOIN Payment Hie       ON P.PhoneID=PaymentID
);
   SELECT * FROM. payment
```

VEL TECH	
EX NO.	5
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	4
RECORD (5)	
TOTAL (20)	14
SIGN WITH DATE	[illegible] 10/5