04/09/25

TASK 5

WRITING JOIN QUERIES EQUIVALENT ANOCOE RECORSIVE QUERIES.

Aim:- To implement and execute join queries equivalent queries and recursive queries using mobile database.

INNER JOIN:-

Returns records that matching values in both tables

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

FROM mobile phones m

INNER JOIN phone specifications

Phone id	brand	model	price
1	Realme	14pro	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_Pd	ram	storage	battery.
1	16GB	256GB	5000 mah
2	8GB	128GB	4500mah
3	12GB	256GB	5500 mah

LEFT (OUTERJOIN) :- Return all records from the table and the matched record from the right table

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

FROM mobile phones m

LEFT JOIN phone, specifications, ON m.phone id = s.phone id

	phone_id	brand	model	price.
1	Realme	Realme	11PRO	30,000
2	Redmi	Redmi	10PRO	15,000
3	VIVO	VIVO	T3PRO	25,000

ram.	storage	battery.
16GB	256GB	5000mah
8GB	128GB	4500mah
12GB	256GB	5000mah

RIGHT (OUTERTION) :- 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 mobile phones m

RIGHT TION phone. specifications.

ON m.phone_id = s.phone_id;

PHONE-Id	brand	model	Price	ram	storage	batter
1	Realme	14Pro	30000	16GB	256GB	5000m
2	Redmi	10Pro	15,000	8GB	128GB	4500m
3.	VIVO	T3 Pro	25,000	12GB	256GB	5500m

FULL OUTER TION:- Return all records when there is a match in either left or right table.

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

FROM Mobile phone m.

FULL OUTER TION phone specification s. ON m. phone id = s. phone_id

Phone-Id	brand	model	Price	ram	storage	battery
1	Realme	14Pro	30000	16GB	256GB	5000 mah
2	Redmi	10Pro	15,000	8GB	128GB	4500 mah
3.	VIVO	T3Pro	25,000	12GB	256GB	5500 mah

1 TION QUERIES

CREATE TABLE

Create table 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;

```
Model VARCHAR (30) NOT NULL;
Price DECIMAL (10,2) CHECK (Price 30,000);
);
CREATE TABLE Purchase (
Purchase ID INT PRIMARY KEY;
UIT ID NOT NULL;
Mobile ID NOT NULL;
Quantity INT Check (Quantity >0);
Purchase Date DATE Default CURRENT DATE;
FOREGIN KEY (UIT ID);
REFERENCES Mobiles (Mobile ID)
);
CREATE TABLE payment (
payment ID INT PRIMARY KEY;
Purchase ID INT UNQUE;
AMOUNT DECIMAL (10,2) NOT NULL,
Payment Date DATE Default
CURRENT - DATE,
Payment method VARCHAR (20)
CHECK ( payment_method INT 'UP' and;
Net banking', '100');
FOREIGN KEY (purchase ID)
REFERENCES Purchase (purchase ID)
```

2. INSERT SAMPLE DATA :-

Insert into Mobile values (', Android item');

(101, 'Realme'),
(102, 'Redmi'),
(102, 'VIVO');

INSERT INTO mobile value payment values

(1, 'Realme'; 101));
(2, 'Redmi'; 102));
(3, 'VIVO'; 101));
(4, 'POCO'; 103));
(5, '1900; 104), -- Invalid phone ID for

OUTER JION example.

INSERT INTO REVIED values

('C1'; 'Database system; 101);
('C2'; 'Good product & worth it', 101);
('C3'; 'product is good' 102),
('C4'; afford to buy it' 103');

INSERT INTO payment values (30,000; 15000,
25 000, 2025-08-11)
1 ROW (rated. completed');

RESULT: Reward inserted sucessfully

3. JOIN QUERIES

a INNER JION

SELET m.phone id, m.brand, m.model, s.ram
s.storage.
s.battery.
FROM mobile Phone m.
INNER JOIN Phone.specification on m.phone-Id=
.s.phone_Id;

b. LEFT JOIN

SELET. m.phone_id, m.brand, m.model s.ram,
s.storage, s.battery
FROM mobile phones.M
LEFT JION Phone on m.phone id = s.phone_id;

(c) RIGHT JION.
SELET 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:-

SELECT: sto.mobile Name, m.model
Name,
FROM mobile Phone
JOIN Brand ON. PHONE ID, m.PHONE ID: USING

subway

```
SELECT mobile Name;
    (SELECT Brand Name FROM Brand, WHERE
    m.phone ID;1 1.phoneID) As model Name
    FROM mobile phone;

RECURSIVE QUERY (PURchase1 iPtriachy)
WITH RECURSIVE purchase1 AS
 SELECT payment ID, PhoneID.
  FROM pre requestfilter
   UNION
   SELECT, Payment ID, PhoneID
   FROM Pre requestfilter P
   TOIN Payment HPreachy ION P.Preveq ID =
    Phone payment ID
   )
   IECECT *FROM, Payment Hieralhy
```

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

11/9

RESULT:- Thus the Implementation of sql commands and using joins recursive queries are executed sucessfully