Task No:5 Writing Join Queries, Equivalent, AND/OR Recursive Queries.

Date:- 09/9/25

Aim:- To implement and execute Join queries, equivalent queries, and recursive queries.

Types of Join in SQL:-

1. Inner Join:- Returns records that have matching values in both tables.

Syntax:- select column-name(s) from table1 INNER JOIN table2 ON table1.column-name = table2.column-name;

2. left outer Join:- Returns all records from the left table, and the matched records from the right table.

Syntax:- select column-name(s) from table1 LEFT JOIN table2 ON table1.column-name = table2.column-name;

3. Right outer Join:- Return all records from the right table, and the matched records from the left table.

Syntax:- select column-name(s) from table1 Right Join table2 on table1.column-name = table2.column-name.

4. Full outer Join:- Returns all records when there is a match in either left or right table.

Syntax:- select column-name(s) from table1 Full outer Join table2 ON table1.column-name = table2.column-name

1. JOIN QUERIES

Create table:-

```
Create table customer (
    customer_ID int primary key,
    name varchar(50),
    address varchar(100)
);

create table bank-account (
    account-number int primary key;
    customer ID int,
    balance int,
    category varchar(50),
```

```
Foreign key (customer ID) references customer (customer ID)
);
create table branch (
        branch ID int primary key,
            branch name varchar (50),
    );
```

2. Insert sample data :-

```
insert into customer (customer ID, name, address) values.
    (101, 'Ram kumar', (chennai);
insert into customer (customer ID, name, address) values.
    (102, 'vijay Rao', 'Hyderabad');
insert into customer (customer ID, name, address) values.
    (103, 'vasu Reddy', 'vizag');
insert into customer (customer ID, name, address) values.
    (104, 'vinay kumar', 'chennai');
insert into customer (customer ID, name, address) values.
    (105, 'Rohit', 'Delhi');
insert into bank-account (account-number, customer ID,
balance, category) values (1001, 101, 15000, 'savings');
insert into bank-account (account-number, customer ID,
balance, category) values (1002, 102, 0, 'current');
Insert into bank-account (account-number, customer ID, balanc
category) values (1003, 103, 5000, 'savings');
insert into bank-account (account-number, customer ID,
balance, Category) values (1004, 105, 2000, 'current');

insert into branch (branch ID, branch Name) values.
    (1, 'chennai Branch');
insert into branch (branch ID, branch Name) values
    (2, 'Hyderabed Branch');
insert into branch (branch IP, branch Name) values.
    (3, 'vizag branch');
```

3. Join Queries:

(a) Inner Join:

Query: select c.name, b.account-number from customer c inner join bank-account b. ON c.customer ID = b.customer ID;

output:

name	account-number
Ram kumar	1001
vijay Rao	1002
vasu Reddy	1003
vinay kuman	1004

(b) left Join:

Query: select c.name, b.account-number from customer left join bank-account b ON c.customerID = b.customerID

output:

name	account-number
Ram kumar	1001
vijay Rao	1002
vasu reddy	1003
vinay kumar	1004
Rohit sharma	NULL

(c) Right joint

Query: select c.name, b.account-number from customer c Right join bank-account b ON c.customerID = b.customerID;

Output:

name	account-number
Ramkumar	1001
vijay Rao	1002
vasu Reddy	1003
vinay kumar	1004

d) full outer Join:

Query: Select c.name, b.account-number from customer c Full outer join bank-account b ON C.customerID=b.customerID;

output:

name	account-number
Ram kumar	1001
vijay Rao	1002
vasu Reddy	1003
vinay kumar	1004
Rohit Sharma	NULL

Equivalent Query:

a) using Join

Query: Select c.name as customer Name, b.account-number as Account Number from customer c Join bank.account b on c.customer ID = b.customer ID;

output:

Customer Name	Account Number.
Ram kumar	1001
vijay Rao	1002
Vasu reddy	1003
Vinay kumar	1004

b) using sub Query

Select c.name as customer name, c select b.account. number From bank-account b where b.customer ID = C.customer ID limit 1) As account Number from Customer c1

Output:

Customer name	Account Number.
Ram kumar	1001
vijay Rao	1002
vasu reddy	1003
vinay kumar	1004
Rohit Sharma	NULL

5. Recursive Query;

Query:- with Recursive Refer Hierorchy As c select customer ID, referred By ID from customer where Reffered by ID is NOT NULL UNION Select C.customer ID, C.ref reffered By from customer C).

Select * From Referral Hierarchy;

output

Customer ID	reffered ID.
102	101
103	102
104	104.

VEL TECH	
EX NO.	5
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	3
RECORD (5)	
OTAL (20)	13
WITH DATE	

10/9

Result:

The implementation of SQL commands using Joins and recursive queries are executed successfully.