Task No: 05 Writing Join Queries, Equivalent, AND/OR
Date: 09/09/25 Recursive Queries

Aim:- To implement and execute join queries, equivalent queries and recursive queries

Types of joins in SQL :-

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

Syntax :- Select column_name(s) from table 1 INNER JOIN table 2 ON table 1. column_name = table 2. column_name;

2. Left Order Join :- Returns all records from the Left table, and the matched records from the right table.

Syntax :- Select column_names from table 1 LEFT JOIN Table2 on table 1. column_name = table 2. column_name.

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

Syntax :- Select column_name(s) from table 1 ~~LEFT~~ RIGHT JOIN table 2 ON table 1. column_name = table 2. column_name.

4. FULL Order Join : Returns all records when there is a match in either left or right table.

Syntax :- Select column_name(s) from table 1 Full outer Join table 2 ON table 1. column_name = table 2. column_name;

1. Join Queries

Create Tables

```
Create table customer(
    CustomerID int primary key,
    name varchar(50),
    address varchar(100), refered by ID INT NULL,
);  Foreign key (referred by ID References. customer (customerID)

create table bank_account (
    account_number int primary key;
    customerID for customer ID int;
    balance int,
    category varchar(50),
```

```
    foreign key (customerID) references customer (customerI
);
create table branch (
    branch_ID int primary key,
    branch_Name varchar(50),
);
```

2. Insert Sample data

```
insert into customer (customerID, name, address) values (101, 'Ram kumar', 'chennai');
insert into customer (customerID, name, address) values (102, 'Vijay Rao', 'Hyderabad');
insert into customer (customerID, name, address) values (103, 'Vasu reddy', 'vizag');
insert into customer (customerID, name, address) values (104, 'Rohit', 'chennai');
insert into customer (customerID, name, address) value (105, 'Vinay kumar', '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, customerID, balance, category) values (1003, 103, 5000, 'Savings');
insert into bank_account (account_number, customerID, balance, Categor
values (1004, 105, 2000, 'current');
insert into branch (branchID, brach Name) values (1, 'chennai Branch'
insert into branch (branchID, branch Name) values (2, 'Hyderabad Bran
insert into branch (branchID, branch Name) values (3, 'vizag Branch'
```

3. Join Queries :-

a) Inner Join :-

Query :- select c.name, b.account_number for customer c in
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

b. Left join :-

Query : select c.name, b.account_number from customer c 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 Join :-

Query : select c.name, b.account_number from customer c Right 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

d. Full Order Join :-

Query :- Select c.name, b.account_number from customer c full order join bank_account b on c.customerID = b.customerID;

Output :-

Name	account_number
Ramkumar	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.customerID = 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 customerName, (select b.account_number From bank_account b where b.customerID = c.customer ID limit 1) As account Number from customer c;

Output

Customer Name	Account Number
Ram kumar	1001
Vijay Rao	1002
vasu reddy	1003
Vinay kumar	1004
Rohit Sharma	NULL

T

5. Recursive Query:

Query: with Recursive Referral Hierarchy AS (select customer ID, referred By ID from customer where referred By ID is NOT NULL UNION

Select c.customerID, C.referedByID From Customer c)

Select * from Referral Hierarchy;

Output :-

Customer ID	referred ID
102	101
103	102
104	103

TECH	
EX NO	5
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	-
TOTAL	15

9/5/25

Result:

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