Task No:- 5 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 table1 INNER JOIN table 2 ON table1.Column_name = table 2. Column_Name;

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

Syntax:- Select Column_name(s) From table1 LEFT JOIN table 2 on table1. Column_name = table 2.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 table 2 on table1. Column_name = table 2. 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 table 2 ON table1.Column_Name = table2. Column_name;

Join Queries

Create Tables :-

```
Create table Customer (
 Customer ID int primary key,
 name varchar (50),
 address varchar (100) reference by ID IN T NULL,
Foreign key (reference ID) Reference customer (Customer ID)
);
Create table bank-account (
  account_number int primary key;
  Customer ID int,
  balance int,
  Category varchar(50),
  foreign key (customer ID) reference 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) value
(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 customer( customer ID,~~
insert into bank-account (account_number, Customer ID,
balance, category) values (1001 ,101, 15000, 'savings');
```

insert into bank_account (account_number, customerID, balance, Category) values (1002, 102, 0, 'current');
insert into bank_account (account_number, customer ID, balance Category) values (1003, 103, 5000, 'savings');
insert into bank_account (account_number, customerID, balance, Category) values (1004, 105, 2000, 'current');

insert into branch (branch ID, branch name) values (1, 'Chennai Branch');
insert into branch (branchID, branch name) values (2, 'Hyderabad Branch');
insert into Branch (BranchID, 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.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.Customer ID = b.customerID;

output

Name	account_number
Ram kumar	1001
Vijay Rao	1002
Vasu Reddy	1003
Vinay kumar	1004
Rohit Sharma	1005

c) Right Join:-

Query:- Select c.name, b.account_number from customer c Right Join bank_account b ON C.CustomerID = b.customer ID;

output:-

Name	account_number
Ram kumar	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.customer ID = b.customerID;

name	account_number
Ram kumar	1001
Vijay Rao	1002
Vasu Reddy	1003
Vinay kumar	1004
Rohit Sharma	1005

Equivalent Query:

a) using Join

Query :- Select c.name AS Customer Name, b.account_Numbe
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

Query :- Select c.name As customer Name, (select b.account_
Number From bank_account b where b.customerID = c. custome
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

5. Recursive Query :-

Query :- with Recursive Refernal iteracy AS (select
Customer ID, reference By ID from customer where
By ID is NOT NULL UNION

Select c.customerID, C.referencebyID From Customer c Join Referral Hierarchy on c.refered by ID = rh.customer ID) Select * from Referral Hierarchy;

output:-

Customer ID	referred by ID
102	101
103	102
104	103

VEL TECH	
EX NO.	8
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	4
VIVA VOCE (5)	5
RECORD (5)	—
TOTAL (20)	15
TH DATE	

[illegible]

Result:- The implementation of SQL Commands using Joins and recursive Queries are executed Successfully