Experiment No.6
Implement various join operations.
Date of Performance:
Date of Submission:

Aim :- Write simple query to implement join operations(equi join, natural join, inner join, outer joins).

Objective :- To apply different types ofjoin to retrieve queries from the database management system.

Theory:

SQL Join statement is used to combine data or rows from two or more tables based on a common field between them. Different types of Joins are as follows:

- INNER JOIN
- LEFT JOIN
- RIGHT JOIN
- FULL JOIN

A. INNER JOIN:

The INNER JOIN keyword selects all rows from both the tables as long as the condition is satisfied. This keyword will create the result-set by combining all rows from both the tables where the condition satisfies i.e value of the common field will be the same.

Syntax:

SELECT table I .columnl ,table I .column2,table2.columnl ,....

FROM tablel

INNER JOIN table2

ON tablel .matching_column = table2.matching column; tablel: First

table.

table2: Second table matching_column: Column common to

both the tables.

B. LEFT JOIN:

This join returns all the rows of the table on the left side of the join and matches rows for the table on the right side of the join. For the rows for which there is no matching row on the right side, the result-set will contain null. LEFT JOIN is also known as LEFT OUTER JOIN.

Syntax:

SELECT tablel .column I ,table I .column2,table2.column I ,....

FROM tablel

LEFT JOIN table2

ON tablel .matching_column = table2.matching column; tablel: First

table.

table2: Second table matching_column: Column common to

both the tables.

C. RIGHT JOIN:

RIGHT JOIN is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of the join. For the rows for which there is no matching row on the left side, the result-set will contain null. RIGHT JOIN is also known as RIGHT OUTER JOIN.

Syntax:

SELECT table I .columnl ,table I .column2,table2.columnl,....

FROM tablel

RIGHT JOIN table2

ON tablel .matching_column = table2.matching column; tablel: First table.

table2: Second table matching_column: Column common to both the tables.

D. FULL JOIN:

FULL JOIN creates the result-set by combining results of both LEFT JOIN and RIGHT JOIN. The result-set will contain all the rows from both tables. For the rows for which there is no matching, the result-set will contain NULL values.

Syntax:

SELECT tablel .column I ,table I .column2,table2.column I ,....

FROM tablel

FULL JOIN table2

ON tablel .matching_column = table2.matching column; tablel: First table.

table2: Second table matching_column: Column common to both the tables.

Implementation:

train table:

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

TrainID	TrainName	RouteID	DepartureTime	ArrivalTime	RouteID	Origin	Destination	Distance	Duration
101	Express 1	1	08:00:00	11:00:00	1	City A	City B	200	3
102	Local 1	2	09:00:00	11:00:00	2	City B	City C	150	2
103	Express 2	3	10:00:00	14:00:00	3	City A	City C	300	4

Department table:

```
USE railway_managment_system;
CREATE TABLE TrainRoutes (
    RouteID INT PRIMARY KEY,
    Origin VARCHAR(50),
    Destination VARCHAR(50),
    Distance FLOAT,
    Duration INT
);

INSERT INTO TrainRoutes (RouteID, Origin, Destination, Distance, Duration)
VALUES
(1, 'City A', 'City B', 200, 3),
(2, 'City B', 'City C', 150, 2),
(3, 'City A', 'City C', 300, 4);

INSERT INTO Trains (TrainID, TrainName, RouteID, DepartureTime, ArrivalTime)
VALUES
(101, 'Express 1', 1, '08:00:00', '11:00:00'),
(102, 'Local 1', 2, '09:00:00', '11:00:00'),
```

1) Inner join:

SELECT trains.train_id, trains.train_name, stations.station_name, trains.departure_time
FROM trains
INNER JOIN stations ON trains.departure_station_id = stations.station_id;

2)

Output

Action Output

#	Time	Action	Message
114	02:11:50	INSERT INTO TrainRoutes (RouteID, Origin, Destination, Distance, Duration) VALUES (1, 'City A', 'City B', 20...	3 row(s) affected Reco
115	02:11:50	INSERT INTO Trains (TrainID, TrainName, RouteID, DepartureTime, ArrivalTime) VALUES (101, 'Express 1', ...	3 row(s) affected Reco
116	02:12:20	SELECT * FROM railway_managment_system.train LIMIT 0, 1000	2 row(s) returned
117	02:12:47	SELECT * FROM railway_managment_system.trainroutes LIMIT 0, 1000	3 row(s) returned
118	02:13:20	SELECT * FROM railway_managment_system.trains LIMIT 0, 1000	3 row(s) returned
119	02:14:10	SELECT * FROM Trains INNER JOIN TrainRoutes ON Trains.RouteID = TrainRoutes.RouteID LIMIT 0, 1000	3 row(s) returned

2)Left join:

SELECT trains.train_id, trains.train_name, stations.station_name, trains.departure_time
FROM trains
LEFT JOIN stations ON trains.departure_station_id = stations.station_id;

3)Right join:

SELECT trains.train_id, trains.train_name, stations.station_name, trains.departure_time
FROM trains
RIGHT JOIN stations ON trains.departure_station_id = stations.station_id;

4)Full join:

SELECT trains.train_id, trains.train_name, stations.station_name, trains.departure_time
FROM trains
FULL OUTER JOIN stations ON trains.departure_station_id = stations.station_id;

Conclusion:

In conclusion, the provided SQL queries illustrate various join operations commonly used in office management scenarios to combine data from multiple tables. Here's a summary of each join type:

1. Inner Join: Retrieves rows from both tables where there is a match between the specified columns, excluding rows where there is no match.

2. Left Outer Join: Retrieves all rows from the left table (first table in the JOIN clause) and matching rows from the right table. If there is no match, NULL values are returned for the columns from the right table.

4. Right Outer Join: Retrieves all rows from the right table (second table in the JOIN clause) and matching rows from the left table. If there is no match, NULL values are returned for the columns from the left table.

Each join type offers a different way to combine data, allowing for flexibility in querying and analyzing data from related tables in office management databases. Depending on the specific requirements of the analysis or report, different join types may be chosen to achieve the desired results efficiently and accurately.

1. Illustrate how to perform natural join for the joining attributes with different names with a suitable example.
2. Illustrate significant differences between natural join equi join and inner join.