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:

Employee table:

Result Grid | Filter Rows: | Edit:

employee_id	employee_name	department_id	salary
1	John Doe	1	50000.00
2	Jane Smith	1	55000.00
3	Michael Johnson	2	48000.00
4	Emily Brown	2	52000.00

Department table:

Result Grid | Filter Rows:

department_id	department_name
1	Finance
2	HR
3	IT

1)Inner join:

```
SELECT e.employee_id, e.employee_name, e.salary, d.department_name
FROM employees e
INNER JOIN departments d ON e.department_id = d.department_id;
```

Result Grid | Filter Rows: | Export:

employee_id	employee_name	salary	department_name
1	John Doe	50000.00	Finance
2	Jane Smith	55000.00	Finance
3	Michael Johnson	48000.00	HR
4	Emily Brown	52000.00	HR

2)Left join:

```
SELECT e.employee_id, e.employee_name, e.salary, d.department_name
FROM employees e
LEFT OUTER JOIN departments d ON e.department_id = d.department_id;
```

Result Grid | Filter Rows: | Export:

employee_id	employee_name	salary	department_name
1	John Doe	50000.00	Finance
2	Jane Smith	55000.00	Finance
3	Michael Johnson	48000.00	HR
4	Emily Brown	52000.00	HR

3)Right join:

```
SELECT e.employee_id, e.employee_name, e.salary, d.department_name
FROM employees e
RIGHT OUTER JOIN departments d ON e.department_id = d.department_id;
```

Result Grid | Filter Rows: | Export:

employee_id	employee_name	salary	department_name
2	Jane Smith	55000.00	Finance
1	John Doe	50000.00	Finance
4	Emily Brown	52000.00	HR
3	Michael Johnson	48000.00	HR
NULL	NULL	NULL	IT

4)Full join:

```
SELECT e.employee_id, e.employee_name, e.salary, d.department_name
FROM employees e
FULL OUTER JOIN departments d ON e.department_id = d.department_id;
```

Result Grid | Filter Rows: | Export:

employee_id	employee_name	salary	department_name
1	John Doe	50000.00	Finance
2	Jane Smith	55000.00	Finance
3	Michael Johnson	48000.00	HR
4	Emily Brown	52000.00	HR
NULL	NULL	NULL	IT

Conclusion:

1. Illustrate how to perform natural join for the joining attributes with different names with a suitable example.

Ans.: Performing a natural join with joining attributes having different names requires explicitly specifying the join condition. Here's a concise example:

Example:

SELECT * FROM Employees

NATURAL JOIN Departments

ON Employees.dept_id = Departments.department_id;

In this example, Employees and Departments tables have different column names (dept_id and department_id). The ON clause specifies the common columns for the natural join.

2. Illustrate significant differences between natural join, equi-join and inner join.

Ans.: Differences Between Natural Join, Equi Join, and Inner Join:

Natural Join: Automatically matches columns with the same name but can produce unexpected results.

Equi Join: Specifies join conditions explicitly, allowing joining attributes with different names.

Inner Join: Returns rows that satisfy the join condition specified in the ON clause, providing control over the join condition.