



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 of join 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 table1 .column1 ,table1 .column2,table2.column1 ,....
```

```
FROM table1
```

```
INNER JOIN table2
```

```
ON table1 .matching_column = table2.matching column; table1: 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 table1 .column1 ,table1 .column2,table2.column1 ,....
```

```
FROM table1
```

```
LEFT JOIN table2
```

```
ON table1 .matching_column = table2.matching column; table1: 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 .column1 ,table I .column2,table2.column1,....

FROM table1

RIGHT JOIN table2

ON table1 .matching_column = table2.matching column; table1: 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 table1 .column1 ,table I .column2,table2.column1 ,....

FROM table1

FULL JOIN table2

ON table1 .matching_column = table2.matching column; table1: First table.

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

Implementation:

Products table:

	product_id	product_name	price	stock_quantity	category
▶	1	Ball	30.00	100	Category 1
	2	Bat	100.00	50	Category 2
	3	Stumps	800.00	300	Category 3
	4	Helmet	850.00	200	Category 4
	5	Kneepads	500.00	700	Category 5
*	NULL	NULL	NULL	NULL	NULL

Supplier table:

	supplier_name	contact_name	contact_email
▶	Sports Supply	Karan Ahuja	karan1234@sportssupply.com
	Homedecor Supply	Karan Singh	karan134@homedecorsupply.com
	Textile Supply	Goldiee Singh	goldie1234@gmail.com

1)Inner join:



```
SELECT p.product_name,p.price,p.stock_quantity
FROM products p
INNER JOIN suppliers s ON p.supplier_name = s.supplier_name;
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
product_name	price	stock_quantity	

2)Left join:

```
SELECT p.product_name,p.price,p.stock_quantity
FROM products p
LEFT OUTER JOIN suppliers s ON p.supplier_name = s.supplier_name;
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
product_name	price	stock_quantity	
Ball	30.00	100	
Bat	100.00	50	
Stumps	800.00	300	
Helmet	850.00	200	
Kneepads	500.00	700	
Ball	30.00	100	
Bat	100.00	50	
Stumps	800.00	300	
Helmet	850.00	200	
Kneepads	500.00	700	

3)Right join:

```
SELECT p.product_name,p.price,p.stock_quantity
FROM products p
RIGHT OUTER JOIN suppliers s ON p.supplier_name = s.supplier_name;
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
product_name	price	stock_quantity	
NULL	NULL	NULL	
NULL	NULL	NULL	
NULL	NULL	NULL	

4)Full join:

```
SELECT p.product_name,p.price,p.stock_quantity
FROM products p
FULL OUTER JOIN suppliers s ON p.supplier_name = s.supplier_name;
```



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 Products
NATURAL JOIN Suppliers
ON Products.supplier_name = Suppliers.supplier_name
```

In this example, Products and Supplier tables have different column names (product_id and supplier_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.