TG ACADEMIC RESOURCE SERIES

SQL PROGRAMMING

Understanding SQL Joins

INNER JOIN, LEFT JOIN, RIGHT JOIN & FULL OUTER JOIN

SUBJECT	SQL / Database Management Systems
LEVEL	BTech / BCA / MCA / BSc / MSc / MTech
RESOURCE TYPE	Sample learning note
PUBLICATION ID	SQL-001 | Version 1.0

Prepared by Tanmoy Gupta

Computer Science Educator & Industry Professional

This sample explains how SQL combines rows from two related tables. A single employee-department example is used to compare matching rows, unmatched rows and the result produced by each major join type.

A focused introduction—not a complete SQL course or a full query-practice workbook.

1. Learning Objectives

After studying this note, you should be able to:

- Explain why relational queries combine rows from multiple tables.
- Distinguish INNER, LEFT, RIGHT and FULL OUTER JOIN.
- Read a join condition written with ON.
- Predict which unmatched rows remain in an outer join.
- Interpret the NULL values produced when no matching row exists.

2. What Is a JOIN?

A **JOIN** combines rows from two tables based on a related column or a join condition. The join condition tells the database which rows are considered a match.

For example, an employee record may store a DeptID, while the department name is stored in a separate DEPARTMENT table. A join can bring the employee and department information into one query result.

3. Sample Tables

EMPLOYEE

EmpID	EmpName	DeptID
101	Asha	D1
102	Ravi	D2
103	Meera	D1
104	Kabir	D3
105	Nita	NULL

DEPARTMENT

DeptID	DeptName
D1	Sales
D2	IT
D4	Finance

Notice the unmatched values: employee 104 refers to D3, which is not in DEPARTMENT; employee 105 has no DeptID; and department D4 has no employee in EMPLOYEE.

Common join condition used in this note:

```
FROM EMPLOYEE e\nJOIN DEPARTMENT d ON e.DeptID = d.DeptID
```

4. INNER JOIN

An **INNER JOIN** returns only rows for which the join condition matches in both tables. Unmatched employees and unmatched departments are excluded.

```
SELECT e.EmpID, e.EmpName, d.DeptName
FROM EMPLOYEE e
INNER JOIN DEPARTMENT d
    ON e.DeptID = d.DeptID;
```

Result of INNER JOIN

EmpID	EmpName	DeptName
101	Asha	Sales
102	Ravi	IT
103	Meera	Sales

Interpretation: Only three employees have a DeptID that matches a row in DEPARTMENT. Kabir, Nita and the Finance department do not appear.

5. LEFT JOIN (LEFT OUTER JOIN)

A **LEFT JOIN** returns every row from the left table (EMPLOYEE) and matching rows from the right table (DEPARTMENT). If there is no match, the right-table columns contain NULL.

```
SELECT e.EmpID, e.EmpName, d.DeptName
FROM EMPLOYEE e
LEFT JOIN DEPARTMENT d
    ON e.DeptID = d.DeptID;
```

Result of LEFT JOIN

EmpID	EmpName	DeptName
101	Asha	Sales
102	Ravi	IT
103	Meera	Sales
104	Kabir	NULL
105	Nita	NULL

Interpretation: All five employees remain in the result. Kabir and Nita have no matching department row, so DeptName is NULL.

6. RIGHT JOIN (RIGHT OUTER JOIN)

A **RIGHT JOIN** returns every row from the right table (DEPARTMENT) and matching rows from the left table (EMPLOYEE). If no employee matches a department, the employee columns contain NULL.

```
SELECT e.EmpID, e.EmpName, d.DeptName
FROM EMPLOYEE e
RIGHT JOIN DEPARTMENT d
    ON e.DeptID = d.DeptID;
```

Result of RIGHT JOIN

EmpID	EmpName	DeptName
101	Asha	Sales
103	Meera	Sales
102	Ravi	IT
NULL	NULL	Finance

Interpretation: All three departments remain. Finance has no matching employee, so EmpID and EmpName are NULL.

7. FULL OUTER JOIN

A **FULL OUTER JOIN** returns all matching rows plus all unmatched rows from both tables. Missing-side columns are filled with NULL.

```
SELECT e.EmpID, e.EmpName, d.DeptName
FROM EMPLOYEE e
FULL OUTER JOIN DEPARTMENT d
    ON e.DeptID = d.DeptID;
```

Conceptual result of FULL OUTER JOIN

EmpID	EmpName	DeptName
101	Asha	Sales
103	Meera	Sales
102	Ravi	IT
104	Kabir	NULL
105	Nita	NULL
NULL	NULL	Finance

The result contains the three matching employee-department rows, both unmatched employees, and the unmatched Finance department.

Database compatibility note

PostgreSQL and SQL Server support FULL OUTER JOIN syntax. MySQL does not support FULL OUTER JOIN directly; a common workaround is to combine a LEFT JOIN with the unmatched right-side rows using UNION ALL and an IS NULL filter.

8. FULL OUTER JOIN Workaround in MySQL

The following pattern returns the left-join result and then appends only right-table rows that had no employee match. The IS NULL test uses the non-nullable employee key EmpID.

```
SELECT e.EmpID, e.EmpName, d.DeptName
FROM EMPLOYEE e
LEFT JOIN DEPARTMENT d
    ON e.DeptID = d.DeptID

UNION ALL

SELECT e.EmpID, e.EmpName, d.DeptName
FROM EMPLOYEE e
RIGHT JOIN DEPARTMENT d
    ON e.DeptID = d.DeptID
WHERE e.EmpID IS NULL;
```

9. Common Mistakes to Avoid

• Confusing the left and right table: LEFT JOIN preserves the table written before LEFT JOIN; RIGHT JOIN preserves the table written after RIGHT JOIN.
• Assuming NULL means zero or an empty string. NULL indicates missing or unknown data in this result.
• Forgetting the join condition, which can produce a Cartesian product or an unintended result.
• Applying a filter to the optional side of an outer join in the WHERE clause without considering that it may remove NULL-extended rows.

A subtle ON vs. WHERE point

With a LEFT JOIN, putting a condition such as d.DeptName = 'Sales' in WHERE filters the final result and removes rows where d.DeptName is NULL. If the intention is to preserve all employees while matching only Sales departments, the filter generally belongs in the ON clause.

10. Check Your Understanding

Q1. Which join returns only rows with a match in both tables?

Q2. Which join should be used when every employee must appear, even if the employee has no matching department?

Q3. In the sample data, how many rows are returned by the FULL OUTER JOIN?

Q4. Why does the Finance row contain NULL for EmpID and EmpName in a RIGHT JOIN?

Q5. Does MySQL accept the FULL OUTER JOIN syntax directly?

Suggested Answers (brief)

A1. INNER JOIN.
A2. LEFT JOIN, with EMPLOYEE as the left table.
A3. Six rows: three matches, two unmatched employees and one unmatched department.
A4. Finance has no matching employee row.
A5. No. Use a LEFT JOIN plus unmatched right-side rows, commonly combined with UNION ALL.

Where to Go Next

A complete SQL joins module can extend into self joins, non-equi joins, joining three or more tables, aggregate queries after joins, anti joins, semi joins and query optimisation. These are separate topics for guided practice.

In guided classes: students can practise predicting result sets, writing queries, debugging join conditions and solving multi-table database problems.

About this resource
TG Academic Resource Series | SQL-001 | Version 1.0
Prepared by Tanmoy Gupta - Computer Science Educator & Industry Professional

Original educational material prepared for student learning and revision. Redistribution or commercial reproduction without permission is not authorised.