

How to identify sql join interview questions use which join they asked to solve

ChatGPT said:

To identify which **SQL JOIN** to use in an interview question, follow this structured approach. Interviewers may not directly mention "INNER JOIN", "LEFT JOIN", etc., but the wording of the question gives strong clues.

🔍 Step-by-Step Method to Identify the JOIN Type

Keyword or Clue in Question	Likely JOIN Type	Explanation
"Return only matching records from both tables"	INNER JOIN	You need common rows that match in both tables.
"Return all records from [Table A], and matching from [Table B]"	LEFT JOIN	Keep all rows from Table A, even if there is no match in B.
"Return all records from [Table B], and matching from [Table A]"	RIGHT JOIN	Same as LEFT JOIN, but keep all from Table B.
"Return all records from both tables, match when possible"	FULL OUTER JOIN	Include all records from both sides.
"Find records in A that do not have a match in B"	LEFT JOIN + WHERE B.col IS NULL	Look for non-matching records.
"Find unmatched records in both tables"	FULL OUTER JOIN + WHERE A.col IS NULL OR B.col IS NULL	Non-matching rows on both sides.
"Self join", or "compare rows in the same table"	SELF JOIN	Join the table with itself.

💡 Examples by Question Type

1. "List employees who have a manager."

- **JOIN Type:** INNER JOIN
- **Why:** You only want employees that match to a manager.

2. "List all employees and their departments (even if no department)."

- **JOIN Type:** LEFT JOIN
- **Why:** All employees, some may not belong to a department.

3. "Find customers who never placed an order."

- **JOIN Type:** LEFT JOIN + WHERE order_id IS NULL
- **Why:** All customers, filter those without orders.

4. "Show all products and all suppliers, even if no match."

- **JOIN Type:** FULL OUTER JOIN
- **Why:** Include both unmatched suppliers and products.

□ Tip to Remember:

- **INNER JOIN = Intersection**
- **LEFT JOIN = All from left + matching from right**
- **RIGHT JOIN = All from right + matching from left**
- **FULL OUTER JOIN = Union with NULLs for missing matches**

☑ Quick Checklist

☑ Question Wording	JOIN Type
◆ "only those that have"	INNER JOIN
◆ "all from this table even if no match"	LEFT JOIN
◆ "all from both even if no match"	FULL OUTER JOIN
◆ "do not have matches"	LEFT JOIN + IS NULL
◆ "join table to itself"	SELF JOIN

Want to practice with sample questions and identify which JOIN to use? I can provide examples.

You said:

Write an sql query to report the first name last name city and state of each person in the person table if the address of a personid is not present in the address table report NULL instead

ChatGPT said:

To solve this problem, you need to:

- Get all people from the Person table.

- Match their address **if available** from the Address table.
- If no matching address exists, show NULL for City and State.

☑ Use LEFT JOIN

Because we want **all persons** even if they don't have an address.

🔗 SQL Query:

sql

CopyEdit

SELECT

 p.FirstName,

 p.LastName,

 a.City,

 a.State

FROM

 Person p

LEFT JOIN

 Address a

ON

 p.PersonId = a.PersonId;

💡 Explanation:

Column From table Description

FirstName Person Always shown

LastName Person Always shown

City Address NULL if no match in Address

State Address NULL if no match in Address

Let me know if you want to practice more such JOIN-based interview questions!

You said:

Yes

ChatGPT said:

Great! 🎉 Here are **5 JOIN-based SQL interview practice questions**, each with hints on what JOIN to use. After each one, you can try solving it, and I'll review your answer if you want.

🔗 Practice Questions

1. Customers Who Never Placed an Order

Tables:

- Customers(CustomerID, Name)
- Orders(OrderID, CustomerID, OrderDate)

□ **Task:** Write a query to return the names of customers who never placed an order.

Hint: Use LEFT JOIN + WHERE Orders.CustomerID IS NULL

2. List All Employees with Their Manager Names

Tables:

- Employees(EmployeeID, Name, ManagerID)

□ **Task:** For each employee, return their name and the name of their manager.

Hint: SELF JOIN — e.ManagerID = m.EmployeeID

3. List Orders with and without Customer Info

Tables:

- Orders(OrderID, CustomerID, Amount)
- Customers(CustomerID, Name)

□ **Task:** List all orders along with customer names. If customer data is missing, show NULL.

Hint: Use LEFT JOIN — some orders might not have valid customer data.

4. Show Students Who Are Not Enrolled in Any Course

Tables:

- Students(StudentID, Name)
- Enrollments(StudentID, CourseID)

□ **Task:** Return names of students with **no enrollments**.

Hint: LEFT JOIN + Enrollments.StudentID IS NULL

5. Combine Product & Supplier Info (All possible data)**Tables:**

- Products(ProductID, Name, SupplierID)
- Suppliers(SupplierID, CompanyName)

□ **Task:** Return all product and supplier info, **even if one doesn't match**.

Hint: Use FULL OUTER JOIN

Common Window Functions and Their Definitions:

Function	Description
ROW_NUMBER()	Assigns a unique sequential number to each row, starting from 1
RANK()	Assigns rank, skips numbers in case of ties (e.g., 1, 2, 2, 4)
DENSE_RANK()	Assigns rank, no skips in case of ties (e.g., 1, 2, 2, 3)
NTILE(n)	Divides rows into n buckets or groups of nearly equal size
LAG()	Returns value from previous row
LEAD()	Returns value from next row
SUM(), AVG()	Running total or average over a moving window (like cumulative total)
FIRST_VALUE()	Gets the first value in the window
LAST_VALUE()	Gets the last value in the window

Self-Introduction: Akash Lenin

Hi, I'm **Akash Lenin**, a passionate and detail-oriented PL/SQL Developer from **Chennai, India**, with hands-on experience in database development and performance optimization. I recently worked as a **Junior Software Analyst at Binary IT Solutions**, where I was actively involved in writing and optimizing PL/SQL code, designing schemas, analyzing large datasets, and tuning queries for business-critical applications.

I hold a **Master's in Computer Applications** from S.A. Engineering College, and my academic foundation is backed by practical exposure to **Oracle 12c, SQL, and PostgreSQL**. During my 6-month industry experience, I developed stored procedures, triggers, views, and ER diagrams, and collaborated closely with senior developers to debug and fine-tune high-volume queries—enhancing performance across modules.

What sets me apart is my **strong grasp of database design, normalization techniques, and execution plan analysis**—which I combine with a learner's mindset and a disciplined approach to coding. I've also completed certifications like *SQL for Data Science* and *Database Management Essentials*, and I'm currently pursuing the **Oracle PL/SQL Developer Certified Associate** credential to strengthen my professional edge.

Beyond work, I enjoy solving real-world database challenges on GitHub and sharing my learnings. I'm now looking to join a forward-thinking team like **Appscore Solutions**, where I can contribute to impactful data projects and grow into a more advanced database engineering role.

Thank you for the opportunity, and I'd be happy to share more about how I can add value to your team.