InterviewBit

SQL Joins Interview Questions

To view the live version of the page, click here.

Contents

SQL Joins Interview Questions for Experienced

(.....Continued)

Let's get Started

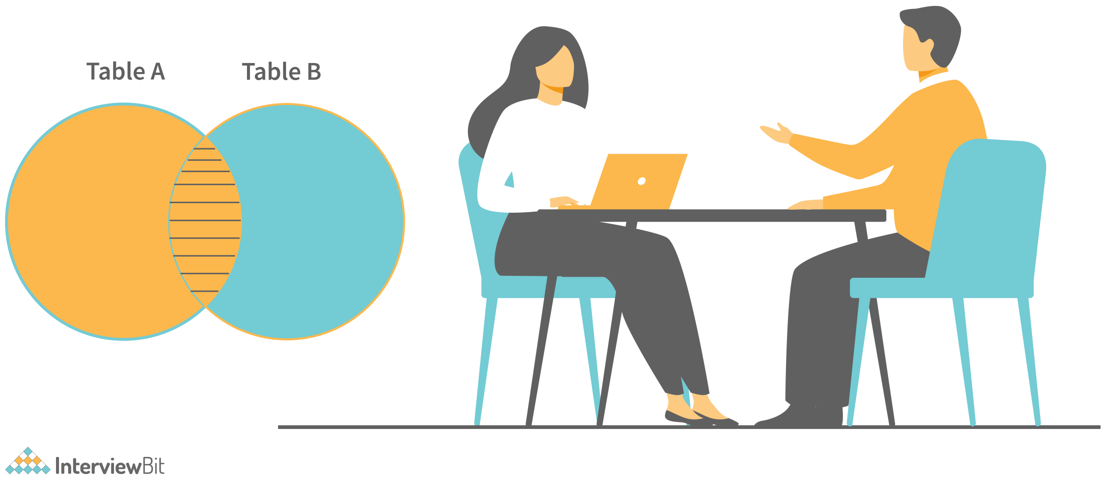
With the rise in demand for data analysts, it is only a matter of time before we begin to see more and more people learning SQL as a necessary skill.

Structured Query Language (SQL) is a special type of language that is meant entirely for storing, searching, updating, manipulating, and retrieving data residing in a table in a relational database.

What if we have two tables that contain different information about the same person, and we would like to display this information on an invoice for that person? In such a case, an SQL join would be necessary. In SQL, Join statements are used to join some or all of the rows from two or more tables based on a common column filed between them. SQL JOIN clauses make a wonderful topic for interviewers to quiz you on during the interview.

SQL Interview Questions on Joins are common in technical interviews, but they can also be asked during data analysts, data scientists, data engineers interviewers, and so forth. Have you ever considered the kinds of SQL JOIN questions you may face during an interview? How prepared are you to answer them?

SQL JOIN INTERVIEW QUESTIONS



In this article, we will cover the most common **SQL Joins interview questions and answers** that have been grouped based on their difficulty level, both for freshers and experienced candidates. To make sure that you become a master at answering these questions, you should practice them repeatedly.

SQL Joins Interview Questions for Freshers

1. What are joins in SQL?

A join clause is a SQL command used to combine records from multiple tables or retrieve data from these tables based on the existence of a common field (column) between them. A join condition and SELECT statement can be used to join the tables. Using the SQL JOIN clause, records can be fetched from two or more tables in a database and combined. In general, they are used when users need to retrieve data from tables that contain many-to-many or one-to-many relationships between them.

Example: Let's take a look at two tables. Here's the Employee table.

Emp_ID	Emp_Name	Emp_No
101	Ashish Kaktan	9450425345
102	Raj Choudhary	8462309621
103	Vivek Oberoi	7512309034
104	Shantanu Khandelwal	9020330023
105	Khanak Desai	8451004522

Here's the Employment table.

Emp_ID	Emp_Profile	Emp_Country	Emp_Join_Date
101	Content Writer	Germany	2021-04-20
104	Data Analyst	India	2022-12-11
105	Software Engineer	India	2022-01-03
108	Development Executive	Europe	2023-02-15
109	Marketing Manager	Mexico	2020-05-23

Let us now join these two tables together using a SELECT statement, as shown below.

```
SELECT Emp_ID, Emp_Name, Emp_No, Emp_Profile, Emp_Country FROM Employee, Employment WHE
```

Output:

Emp_ID	Emp_Name	Emp_No	Emp_Profile	Er
101	Ashish Kaktan	9450425345	Content Writer	Ge
104	Shantanu Khandelwal	9020330023	Data Analyst	In
105	Khanak Desai	8451004522	Software Engineer	In

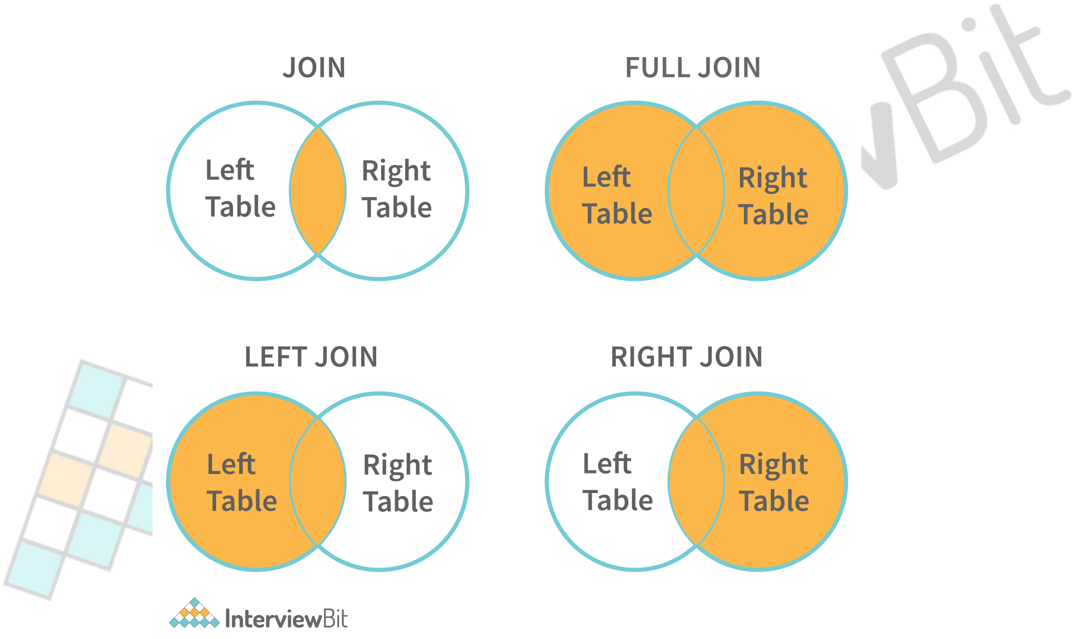
2. What is the importance of SQL joins in database management?

SQL joins are important in database management for the following reasons:

- A method of stitching a database back together to make it easier to read and use.
- Additionally, they maintain a normalized database. Data normalization helps us keep data redundancy low so that when we delete or update a record, we will have fewer data anomalies in our application.
- Joins have the advantage of being faster, and as a result, are more efficient.
- It is almost always faster to retrieve the data using a join query rather than one that uses a subquery.
- By utilizing joins, it is possible to reduce the workload on the database. For example, instead of multiple queries, you can use one join query. So, you can better utilize the database's ability to search, filter, sort, etc.

3. What are the different types of Joins in SQL?

There are various types of join statements you can use depending on the use case you have. Specifically, there are four types of joins as follows:



- **Inner Join:** This method returns datasets that have the same values in both tables.
- **Full Join:** This is also referred to as a full outer join. It combines all rows from the left table and the right table to create the result set. It means that a query would return records from both tables, even if they had NULL values. If there are no matching rows, the result-set (joined table) will display NULL values.
- **Right Join:** This is also referred to as the Right outer join. All records from the right table will be returned, as well as any records that match from the left table.
- **Left Join:** This is also referred to as Left outer join. All records from the left table will be returned, as well as any records that match from the right table.

4. Explain merge join in SQL.

Merge join produces a single output stream resulting from the joining of two sorted datasets using an INNER, FULL, or LEFT join. It is the most effective of all the operators for joining data. Specifically, merge join requires that both inputs be sorted as well as matching meta-data in the joined columns. Users can't join columns of different data types together. Users are not permitted to combine a column with a numeric data type with a column with a character data type.

5. State the difference between inner join and left join.

- **Inner Join:** This join generates datasets that contain matching records in both tables (left and right). By using an inner join, only the rows that match between each of the tables are returned; all non-matching rows are removed.

Example: Let's take a look at two tables. Here's the Tb1_Employee table.

Emp_ID	Emp_Name	Emp_No
101	Ashish Kaktan	9450425345
102	Raj Choudhary	8462309621
103	Vivek Oberoi	7512309034
104	Shantanu Khandelwal	9020330023
105	Khanak Desai	8451004522

Here's the Tb2_Employment table.

Emp_ID	Emp_Profile	Emp_Country	Emp_Join_Date
101	Content Writer	Germany	2021-04-20
104	Data Analyst	India	2022-12-11
105	Software Engineer	India	2022-01-03
108	Development Executive	Europe	2023-02-15
109	Marketing Manager	Mexico	2020-05-23

Let’s perform INNER JOIN on these two tables using a SELECT statement, as shown below:

```
SELECT Emp_Name, Emp_No, Emp_Profile, Emp_Country, Emp_Join_Date
FROM Tb1_Employee INNER JOIN Tb2_Employment
ON Tb1_Employee.Emp_ID=Tb2_Employment.Emp_ID;
```

Output:

Emp_Name	Emp_No	Emp_Profile	Emp_Country
Ashish Kaktan	9450425345	Content Writer	Germany
Shantanu Khandelwal	9020330023	Data Analyst	India
Khanak Desai	8451004522	Software Engineer	India

- **Left Join:** It returns datasets that have matching records in both tables (left and right) plus non-matching rows from the left table. By using a left join, all the records in the left table plus the matching records in the right table are returned.

Example: Let's now perform LEFT JOIN on these two tables using a SELECT statement, as shown below:

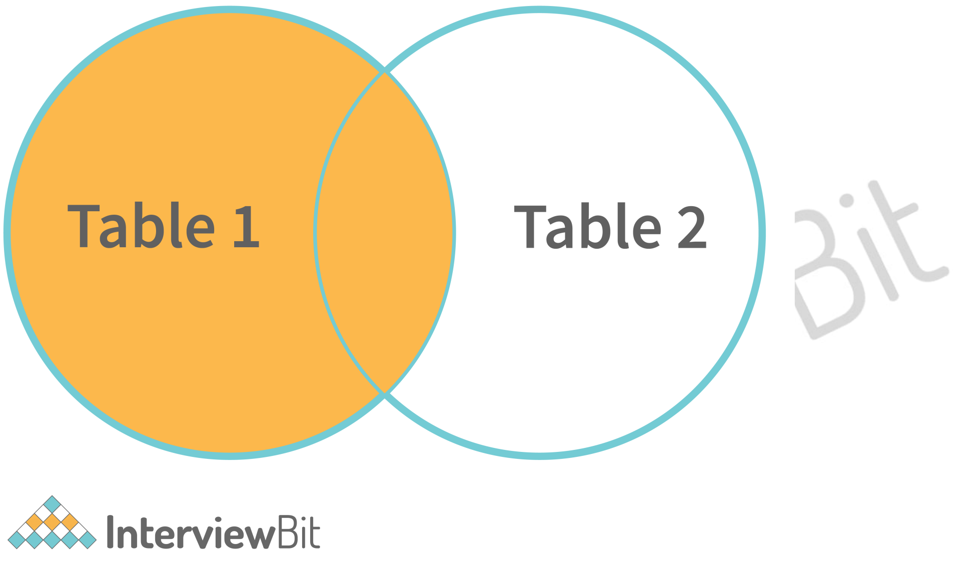
```
SELECT Tb1_Employee.Emp_Name, Tb1_Employee.Emp_No, Tb2_Employment.Emp_Profile, Tb2_Empl
FROM Tb1_Employee LEFT JOIN Tb2_Employment
ON Tb1_Employee.Emp_ID=Tb2_Employment.Emp_ID;
```

Output:

Emp_Name	Emp_No	Emp_Profile	Emp_Country
Ashish Kaktan	9450425345	Content Writer	Germany
Raj Choudhary	8462309621	Null	Null
Vivek Oberoi	7512309034	Null	Null
Shantanu Khandelwal	9020330023	Data Analyst	India
Khanak Desai	8451004522	Software Engineer	India

6. State difference between left join and right join.

- **Left Join:** It returns datasets that have matching records in both tables (left and right) plus non-matching rows from the left table. By using a left join, all the records in the left table plus the matching records in the right table are returned.



Example: Let's take a look at two tables. Here's the Tb1_Employee table.

Emp_ID	Emp_Name	Emp_No
101	Ashish Kaktan	9450425345
102	Raj Choudhary	8462309621
103	Vivek Oberoi	7512309034
104	Shantanu Khandelwal	9020330023
105	Khanak Desai	8451004522

Here's the Tb2_Employment table.

Emp_ID	Emp_Profile	Emp_Country	Emp_Join_Date
101	Content Writer	Germany	2021-04-20
104	Data Analyst	India	2022-12-11
105	Software Engineer	India	2022-01-03
108	Development Executive	Europe	2023-02-15
109	Marketing Manager	Mexico	2020-05-23

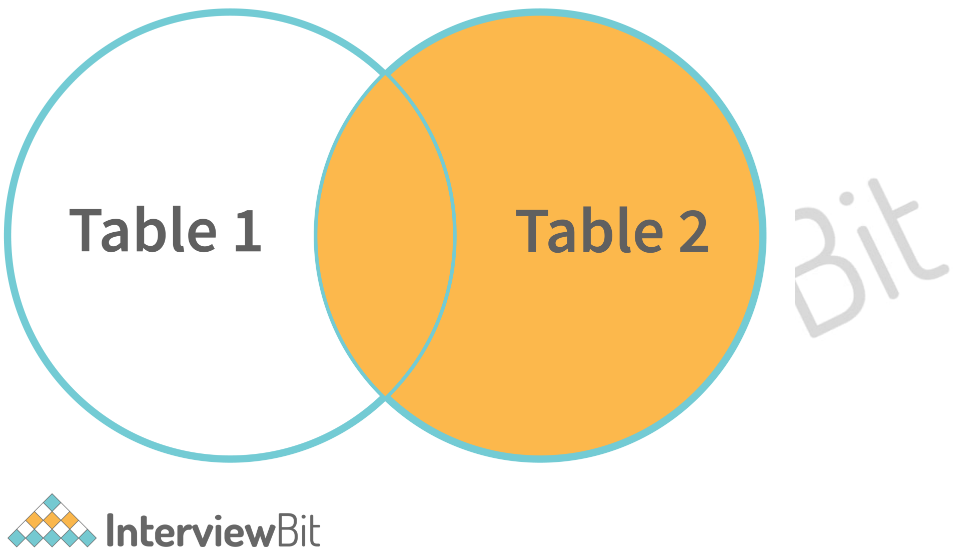
Let's now perform LEFT JOIN on these two tables using a SELECT statement, as shown below:

```
SELECT Tb1_Employee.Emp_Name, Tb1_Employee.Emp_No, Tb2_Employment.Emp_Profile, Tb2_Empl
FROM Tb1_Employee LEFT JOIN Tb2_Employment
ON Tb1_Employee.Emp_ID=Tb2_Employment.Emp_ID;
```

Output:

Emp_Name	Emp_No	Emp_Profile	Emp_Join_Date
Ashish Kaktan	9450425345	Content Writer	2021-04-20
Raj Choudhary	8462309621	Null	Null
Vivek Oberoi	7512309034	Null	Null
Shantanu Khandelwal	9020330023	Data Analyst	2023-02-15
Khanak Desai	8451004522	Software Engineer	2020-05-23

- **Right Join:** It returns datasets that have matching records in both tables (left and right) plus non-matching rows from the right table. By using a left join, all the records in the right table plus the matching records in the left table are returned.



Example: Let’s now perform RIGHT JOIN on these two tables using a SELECT statement, as shown below:

```
SELECT Tb1_Employee.Emp_Name, Tb1_Employee.Emp_No, Tb2_Employment.Emp_Profile, Tb2_Empl
FROM Tb1_Employee RIGHT JOIN Tb2_Employment
ON Tb1_Employee.Emp_ID=Tb2_Employment.Emp_ID;
```

Output:

Emp_Name	Emp_No	Emp_Profile	Emp_Join_Date
Ashish Kaktan	9450425345	Content Writer	2021-04-20
Shantanu Khandelwal	9020330023	Data Analyst	2022-12-11
Khanak Desai	8451004522	Software Engineer	2022-01-03
Null	Null	Development Executive	2023-02-15
Null	Null	Marketing Manager	2020-05-23

7. What is a hash join in SQL?

Just like any other join, the hash join requires two inputs, which are the probe input (inner table) and the build input (outer table). A hash join involves the use of a hash table to identify rows matching between two tables. The hash join is the option when no other join is preferred (possibly due to the absence of sorting or indexing etc). Hash joins are best when joining large data sets that are unsorted and non-indexed.

8. Can you explain nested join in SQL?

A JOIN is one of the mechanisms that we use to combine the data of more than one table in a relational database, and a Nested Join is one of the simplest methods involving the physical joining of two tables. In essence, a Nested Join uses one joining table as an outer input table while the other one serves as an inner input table. With a Nested Loop Join, one row from the outer table is retrieved and then the row is searched for in the inner table; this process is repeated until all the output rows from the outer table have been searched for in the inner table. Nested Loop Join may further be sub-categorized into Indexed Nested, Naive Nested and Temporary Index Nested Loop Join.

9. Write an SQL query to join three tables.

At times, you might need to retrieve data from three or more tables at once. A multi-table join requires consecutive JOIN operations: the first and second tables are joined to form a virtual table and then the third table is joined to this virtual table. Let's take a look at three tables.

Here's the Employee table.

Emp_ID	Emp_Name	Emp_No
101	Ashish Kaktan	9450425345
102	Raj Choudhary	8462309621
103	Vivek Oberoi	7512309034
104	Shantanu Khandelwal	9020330023
105	Khanak Desai	8451004522

Here's the Employment table.

Emp_ID	Emp_Profile	Emp_Email
101	Content Writer	ashish@scaler.com
104	Data Analyst	shantanu@scaler.com
105	Software Engineer	khanak@scaler.com
109	Development Executive	akshay@scaler.com
108	Marketing Manager	nikita@scaler.com

Here's the EmpDetail.

Emp_Country	Emp_Email	Emp_JoinDate
Germany	ashish@scaler.com	2021-04-20
India	shantanu@scaler.com	2022-12-11
India	khanak@scaler.com	2022-01-03
Europe	akshay@scaler.com	2023-02-15
Mexico	nikita@scaler.com	2020-05-23

```
SELECT Emp_Name, Emp_No, Emp_Profile, Emp_Country, EmpJoinDate,
FROM Employee e INNER JOIN Employment m ON
e.Emp_ID = m.EMP_ID INNER JOIN EmpDetail d on
d.Emp_Email = m.Emp_Email;
```

Output:

Emp_Name	Emp_No	Emp_Profile	Emp_Country
101	9450425345	Content Writer	Germany
104	9020330023	Data Analyst	India
105	8451004522	Software Engineer	India

10. How can you join a table to itself?

Another type of join in SQL is a SELF JOIN, which connects a table to itself. In order to perform a self-join, it is necessary to have at least one column (say X) that serves as the primary key as well as one column (say Y) that contains values that can be matched with those in X. The value of Column Y may be null in some rows, and Column X need not have the exact same value as Column Y for every row.

Example: Consider the table Employees.

Emp_ID	Emp_Name	Emp_Profile	Emp_Country
101	Ashish Kaktan	Content Writer	Germany
104	Raj Choudhary	Data Analyst	India
105	Vivek Oberoi	Software Engineer	India
108	Shantanu Khandelwal	Development Executive	Europe
109	Khanak Desai	Marketing Manager	Mexico

For instance, we might wish to display results that only include employees with their managers. By using table aliases and a self-join, this can be accomplished easily.

```
SELECT e.Emp_ID, e.Emp_Name, m.FullName as ManagerName
FROM Employees e JOIN Employees m ON e.ManagerId = m.Emp_ID
```

Output:

Emp_ID	Emp_Name	ManagerName
101	Ashish Kaktan	Raj Choudhary
104	Raj Choudhary	Shantanu Khandelwal
105	Vivek Oberoi	Ashish Kaktan
108	Shantanu Khandelwal	Ashish Kaktan
109	Khanak Desai	Null

SQL Joins Interview Questions for Experienced

11. How should data be structured to support Join Operations in a one-to-many relationship?

Among the most common types of database relationships are the ones involving one-to-many relationships. Consider the example of the Customer and Mobile table.

Customer Table:

Customer_ID	First_Name	Last_Name	Email
1	Sasha	Frank	sashafrank@gmai
2	Isha	Joshi	ishajoshi@gmail.c
3	Danish	Khan	danishkhan@gma

Mobile Table:

Mobile_ID	Customer_ID	OrderPlaced_Date
1	1	2022-03-24
2	3	2021-05-22
3	2	2022-01-05
4	1	2020-10-14
5	2	2021-08-29

A customer can own many mobile phones, but a mobile belongs to one customer only. Therefore, we've defined the Foreign Key column (Customer_ID) in the Mobile table allowing us to perform SELECT queries with JOIN clauses fairly easily.

12. How should data be structured to support Join Operations in a many-to-many relationship?

The many-to-many relationship is a bit more complex than the one-to-many relationship.

Example: Consider a Student table and a Class table, both of which have many-to-many relationships: any student can be enrolled in many classes, and any class can have many students. It means an individual student may have many classes, and there may be many students in each class.

Student Table

Student_ID	Student_Name
1	Asha Bisht
2	Rohit Sharma
3	Karan Tacker

Class Table

Class_ID	Class_Name
1	Maths
2	Science
3	English
4	Physical Education
5	Computer Science

In this case, we cannot add the primary key of one table to that of another, or to both of them, as this only stores a single relationship, while what we really need is multiple relationships. Thus, we use a concept called a bridging table or joining table. The joining tables are those that are placed between two other tables in a many-to-many relationship and are intended to hold a record for each combination of the two other tables. It may seem like quite a chunk of work, but the process is simple and provides a much better data structure. In this case, we will create a new table called ClassEnroll maintaining two columns, one for each of the primary keys of the other table. Those columns store separate records for every class and student combination.

ClassEnroll Table

Student_ID	Class_ID
1	1
1	3
2	4
2	2
3	4
3	1

13. Explain natural join.

Natural Joins are a type of join that combines tables based on columns that share the same name and have the same datatype. Ideally, there should be a common attribute (column) among two tables in order to perform a natural join.

Syntax:

```
SELECT * FROM TableName1 NATURAL JOIN TableName2;
```

Example: Consider two tables Employee and Employment.

Employee

Emp_ID	Emp_Name
1	Khushboo Ahuja
2	Kartik Sharma
3	Milli Desai

Employment

Emp_ID	Emp_Profile
1	Content Writer
2	Business Development Executive
3	Marketing Manager

Now, consider the following query.

```
SELECT * FROM Employee NATURAL JOIN Employment;
```

Output:

Emp_ID	Emp_Name	Emp_Profile
1	Khushboo Ahuja	Content Writer
2	Kartik Sharma	Business Development Executive
3	Milli Desai	Marketing Manager

14. What is an Equi Join?

Equi Joins are a type of INNER Joins where a join is performed between two or more tables using a common column between them. Using the equality sign (=), it compares the data in two columns, if the data is the same, it retrieves it.

Syntax:

```
SELECT * FROM Table1, Table2
WHERE Table1.ColumnName = Table2.ColumnName;
```

OR

```
SELECT column_list  FROM Table1, Table2
WHERE Table1.ColumnName = Table2.ColumnName;
```

OR

```
SELECT * FROM Table1 JOIN Table2 ON Table1.ColumnName = Table2.ColumnName;
Example: Consider the tables Employee and State.
```

Employee Table

Emp_Name	State_ID
Asha Bisht	1
Rohit Sharma	1
Karan Tacker	2
Karan Oberoi	3
Nikhil Bhardwawaj	3

State Table

State_ID	State_Name
1	Madhya Pradesh
2	Bangalore
3	Uttarakhand
4	Rajasthan

Now, lets perform Equi-join using the equality operation and the WHERE clause as follows;

```
SELECT Emp_Name, State_Name FROM Employee, State WHERE Employee.State_ID = State.State_
```

Output:

Emp_Name	State_Name
Asha Bisht	Madhya Pradesh
Rohit Sharma	Madhya Pradesh
Karan Tacker	Bangalore
Karan Oberoi	Uttarakhand
Nikhil Bharadwaj	Uttarakhand

15. What is a Non-Equi Join?

A Non-Equi join entails pulling data from multiple tables by using an INNER join. This type of join matches the columns of two tables based on an inequality using operators such as <, <=, >, >=, !=, BETWEEN, etc.

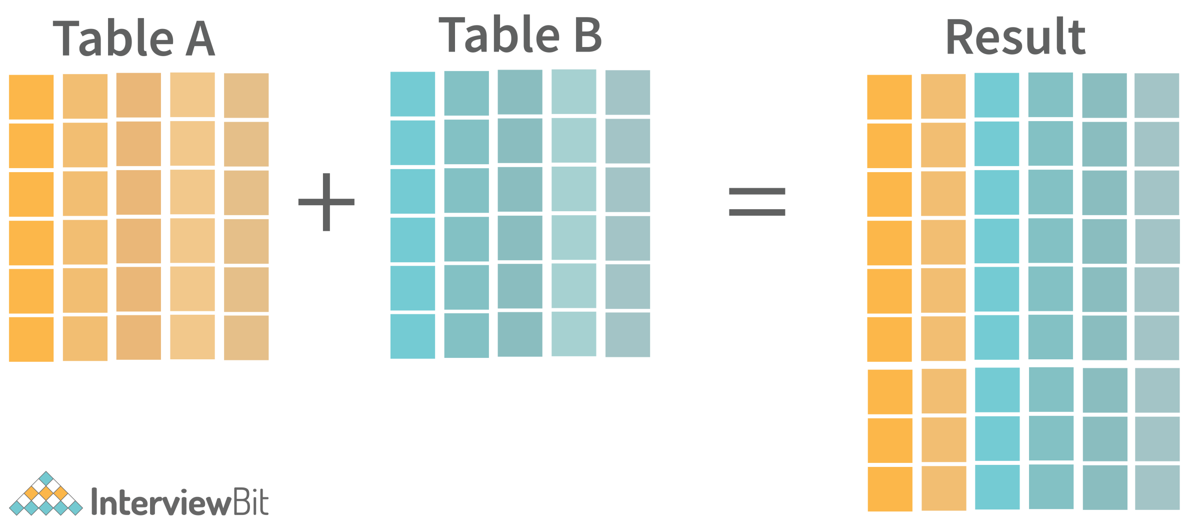
Syntax:

```
SELECT *  FROM TableName1, TableName2
WHERE TableName1.columnName [> | < |  >= | <= | != | BETWEEN ] TableName2.columnName;
```

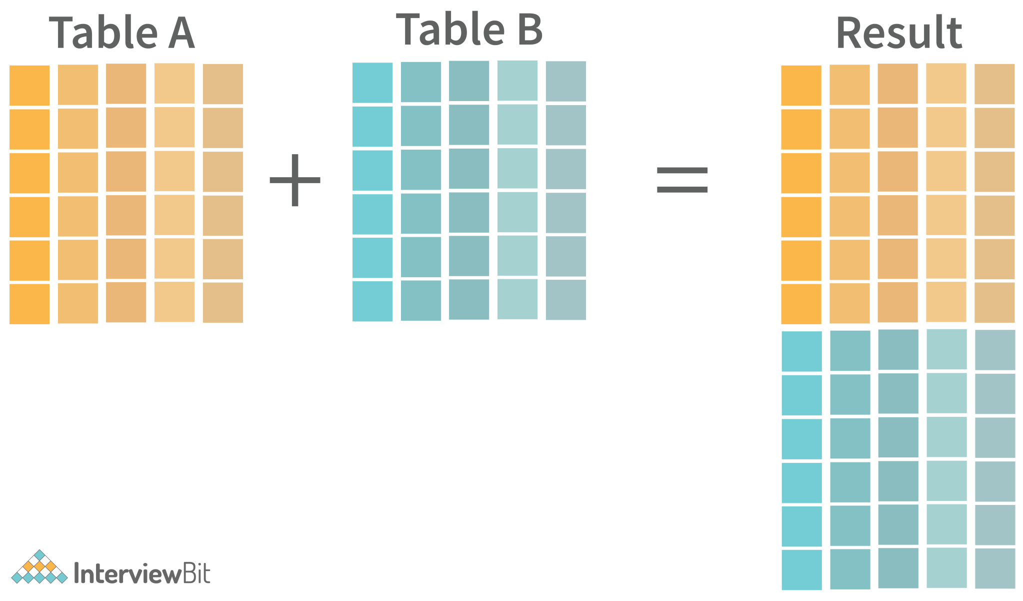
16. What makes a Union clause different from a Join clause?

To combine data from two or more tables, you may use joins and UNIONS in your queries.

- **Join:** Essentially, joins allow you to combine data into new columns horizontally. A join clause is an SQL command used in order to combine records from multiple tables or retrieve data from these tables based on the existence of a common field (column) between them. In JOIN, the columns of the joining tables may differ. Here is a visual representation of how a join looks. See how the columns from Tables A and B have been combined to generate the result.



- **Union:** In SQL, the term UNION is used to combine the results of more than one SELECT statement. Union allows you to combine data into new rows vertically. UNION requires that all queries have the same number of columns and order of columns. Here is a visual representation of how a union looks. See how the columns from Tables A and B have been combined to generate the result.



17. Is it required that the Join condition be based on equality?

No, joins can have non-equi conditions as well. Join clauses can be used with common comparison operators, such as <, <=, >, >=, !=, BETWEEN. For example, listing records, listing unique pairs, and identifying duplicate records are just a few situations where non-equi joins can prove to be useful.

18. State difference between Full Join and Cross Join.

- **Cross Join:** A Cross Join is also referred to as a Cartesian join or Cartesian product. The result set of a cross join is equal to all of the rows from the first table multiplied by all of the rows from the second table. It applies to all columns.

Syntax:

```
SELECT * FROM Table_1 CROSS JOIN Table_2;
```

In the case of two lists, one consisting of 4, 5, 6, and the other consisting of a, b, and c, the Cartesian product between the two lists would be as follows:

4a, 5b, 6c

4a, 5b, 6c

4a, 5b, 6c

- **Full Join:** This is also referred to as a full outer join. In a full outer join, all rows of tables are combined to form a result set. It basically means that a query would return a result set from both tables, even if they had NULL values. A result-set (joined table) will display NULL values if both tables do not contain any matching rows.

Syntax:

```
SELECT * FROM Table1 FULL OUTER JOIN Table2 ON Table1.column_name = Table2.column_name;
```

19. Explain CTE (Common Table Expression) SQL.

Generally, Common Table Expressions (CTEs) are temporary named result sets that you can use to refer to within an UPDATE, INSERT, SELECT, or DELETE statement. The CTEs can be defined by adding a WITH before an UPDATE, INSERT, DELETE, SELECT, OR MERGE statement. Several CTEs may be used within the WITH clause, separated by commas.

20. Distinguish between nested subquery, correlated subquery, and join operation.

- **Join Operation:** A join operation is a binary operation that combines the data or rows from two or more tables based on a field they have in common. There are different types of joins, such as INNER JOIN, FULL JOIN, LEFT JOIN, RIGHT JOIN, etc.
- **Subquery:** A query can be enclosed within another query, so the outer query is called the main query, and the inner query is called a subquery.
    - **Nested Query:** Nested queries execute the inner query first, and only once. An outer query is then executed based on the results of the inner query. The inner query is therefore used to execute the outer query.
    - **Correlated Query:** The outer query is executed first, and for every row of the outer query, the inner query is executed. Thus, values from the outer query are used in the inner query.

21. State difference between Cross join and Natural Join.

- **Natural Join:** Natural Joins are a type of join that combines tables based on columns that share the same name and have the same datatype. Ideally, there should be a common attribute (column) among two tables in order to perform a natural join.
- **Cross Join:** A Cross Join is also referred to as a Cartesian join or Cartesian product. The result set of a cross join is equal to all of the rows from the first table multiplied by all of the rows from the second table. It applies to all columns.

Conclusion

SQL joins are a simple concept that everyone should be able to grasp, even if writing codes and handling them isn't hard. There is only one decision to make: which join to use. What you need to know is what join you should use, which is what makes joining so challenging at times. If you wish to become better at making such decisions, you should gain as much experience as possible, following which you will be more prepared to read and comprehend SQL Joins, understand data, and decide which join type to use. In this article, you will find a number of frequently asked SQL Joins interview questions and answers that will help you prepare convincing answers.

At the end of the day, you should know that the goal is to provide a concise, yet well-illustrated answer that demonstrates your knowledge of the subject matter without going on and on. The best approach would be for you to give an example of a use case you have encountered while using this feature. Practice mock interviews and solve SQL interview questions before your actual interview, as this will boost your confidence and help you prepare for the different scenarios you may face. As previously noted, they are not attempting to certify you, but rather gauge your level of proficiency within your field. In that regard, bear in mind that you should act confidently, professionally, and courteously during the interview. Remember, interviews can be long and tiresome, so do not forget to smile!

Recommended Resources

- https://www.interviewbit.com/sql-interview-questions/
- https://www.interviewbit.com/blog/characteristics-of-sql/
- https://www.interviewbit.com/blog/sql-projects/
- https://www.interviewbit.com/sql-mcq/

Links to More Interview Questions

C Interview Questions	Php Interview Questions	C Sharp Interview Questions
Web Api Interview Questions	Hibernate Interview Questions	Node Js Interview Questions
Cpp Interview Questions	Oops Interview Questions	Devops Interview Questions
Machine Learning Interview Questions	Docker Interview Questions	Mysql Interview Questions
Css Interview Questions	Laravel Interview Questions	Asp Net Interview Questions
Django Interview Questions	Dot Net Interview Questions	Kubernetes Interview Questions
Operating System Interview Questions	React Native Interview Questions	Aws Interview Questions
Git Interview Questions	Java 8 Interview Questions	Mongodb Interview Questions
Dbms Interview Questions	Spring Boot Interview Questions	Power Bi Interview Questions
Pl Sql Interview Questions	Tableau Interview Questions	Linux Interview Questions
Ansible Interview Questions	Java Interview Questions	Jenkins Interview Questions