Top 50 SQL Query Interview Questions and Answers for Practice

Last updated on April 17, 2024

Hello friends! in this post, we will see some of the most common SQL queries asked in interviews. Whether you are a DBA, developer, tester, or data analyst, these **SQL query interview questions and answers** are going to help you.

In fact, I have been asked most of these questions during interviews in the different phases of my career.

If you want to skip the basic questions and start with some tricky SQL queries then you can directly move to our SQL queries interview questions for the experienced section.

Consider the below two tables for reference while trying to solve the **SQL queries for practice**.

Table – EmployeeDetails

EmpId	FullName	ManagerId	DateOfJoining	City
121	John Snow	321	01/31/2019	Toronto
321	Walter White	986	01/30/2020	California
421	Kuldeep Rana	876	27/11/2021	New Delhi

Table – EmployeeSalary

EmpId	Project	Salary	Variable
121	P1	8000	500

321	P2	10000	1000
421	P1	12000	0

For your convenience, I have compiled the top 10 questions for you. You can try solving these questions and click on the links to go to their respective answers.

1. SQL Query to fetch records that are present in one table but not in another table.
2. SQL query to fetch all the employees who are not working on any project.
3. SQL query to fetch all the Employees from EmployeeDetails who joined in the Year 2020.
4. Fetch all employees from EmployeeDetails who have a salary record in EmployeeSalary.
5. Write an SQL query to fetch a project-wise count of employees.
6. Fetch employee names and salaries even if the salary value is not present for the employee.
7. Write an SQL query to fetch all the Employees who are also managers.
8. Write an SQL query to fetch duplicate records from EmployeeDetails.
9. Write an SQL query to fetch only odd rows from the table.
10. Write a query to find the 3rd highest salary from a table without top or limit keyword.

Or, you can also jump to our below two sections on SQL query interview questions for freshers and experienced professionals.

Content

SQL Query Interview Questions for Freshers

Here is a list of top SQL query interview questions and answers for fresher candidates that will help them in their interviews. In these queries, we will focus on

the basic SQL commands only.

1. Write an SQL query to fetch the EmpId and FullName of all the employees working under the Manager with id – '986'.

We can use the EmployeeDetails table to fetch the employee details with a where clause for the manager-

```
SELECT  EmpId, FullName

FROM EmployeeDetails

WHERE ManagerId = 986;
```

2. Write an SQL query to fetch the different projects available from the EmployeeSalary table.

While referring to the EmployeeSalary table, we can see that this table contains project values corresponding to each employee, or we can say that we will have duplicate project values while selecting Project values from this table.

So, we will use the distinct clause to get the unique values of the Project.

```
SELECT DISTINCT(Project)

FROM EmployeeSalary;
```

3. Write an SQL query to fetch the count of employees working in project 'P1'.

Here, we would be using aggregate function count() with the SQL **where** clause-

```
SELECT COUNT(*)

FROM EmployeeSalary

WHERE Project = 'P1';
```

4. Write an SQL query to find the maximum, minimum, and average salary of the employees.

We can use the aggregate function of SQL to fetch the max, min, and average values-

```
SELECT Max(Salary),
```

```
Min(Salary),

AVG(Salary)

FROM EmployeeSalary;
```

5. Write an SQL query to find the employee id whose salary lies in the range of 9000 and 15000.

Here, we can use the ‘Between’ operator with a where clause.

```
SELECT EmpId, Salary

FROM EmployeeSalary

WHERE Salary BETWEEN 9000 AND 15000;
```

6. Write an SQL query to fetch those employees who live in Toronto and work under the manager with ManagerId – 321.

Since we have to satisfy both the conditions – employees living in ‘Toronto’ and working in Project ‘P2’. So, we will use the AND operator here-

```
SELECT EmpId, City, ManagerId

FROM EmployeeDetails

WHERE City='Toronto' AND ManagerId='321';
```

7. Write an SQL query to fetch all the employees who either live in California or work under a manager with ManagerId – 321.

This interview question requires us to satisfy either of the conditions – employees living in ‘California’ and working under Manager with ManagerId – 321. So, we will use the OR operator here-

```
SELECT EmpId, City, ManagerId

FROM EmployeeDetails

WHERE City='California' OR ManagerId='321';
```

8. Write an SQL query to fetch all those employees who work on Projects other than P1.

Here, we can use the NOT operator to fetch the rows which are not satisfying the given condition.

```
SELECT EmpId

FROM EmployeeSalary

WHERE NOT Project='P1';
```

Or using the ‘not equal to’ operator-

```
SELECT EmpId

FROM EmployeeSalary

WHERE Project <> 'P1';
```

For the difference between NOT and <> SQL operators, check this link – Difference between the NOT and != operators.

9. Write an SQL query to display the total salary of each employee adding the Salary with Variable value.

Here, we can simply use the ‘+’ operator in SQL.

```
SELECT EmpId,

Salary+Variable as TotalSalary

FROM EmployeeSalary;
```

10. Write an SQL query to fetch the employees whose name begins with any two characters, followed by a text “hn” and ends with any sequence of characters.

For this question, we can create an SQL query using like operator with ‘_’ and ‘%’ wild card characters, where ‘_’ matches a single character and ‘%’ matches ‘0 or multiple characters.

```
SELECT FullName

FROM EmployeeDetails

WHERE FullName LIKE ‘__hn%’;
```

11. Write an SQL query to fetch all the EmpIds which are present in either of the tables – 'EmployeeDetails' and 'EmployeeSalary'.

In order to get unique employee ids from both tables, we can use the Union clause which can combine the results of the two SQL queries and return unique rows.

```
SELECT EmpId FROM EmployeeDetails

UNION

SELECT EmpId FROM EmployeeSalary;
```

12. Write an SQL query to fetch common records between two tables.

SQL Server – Using INTERSECT operator-

```
SELECT * FROM EmployeeSalary

INTERSECT

SELECT * FROM ManagerSalary;
```

MySQL – Since MySQL doesn't have INTERSECT operator so we can use the subquery-

```
SELECT *

FROM EmployeeSalary

WHERE EmpId IN

(SELECT EmpId from ManagerSalary);
```

13. Write an SQL query to fetch records that are present in one table but not in another table.

SQL Server – Using MINUS- operator-

```
SELECT * FROM EmployeeSalary

MINUS

SELECT * FROM ManagerSalary;
```

MySQL – Since MySQL doesn't have a MINUS operator so we can use LEFT join-

```
SELECT EmployeeSalary.*

FROM EmployeeSalary

LEFT JOIN

ManagerSalary USING (EmpId)

WHERE ManagerSalary.EmpId IS NULL;
```

14. Write an SQL query to fetch the EmpIds that are present in both the tables – 'EmployeeDetails' and 'EmployeeSalary.

Using subquery-

```
SELECT EmpId FROM

EmployeeDetails

where EmpId IN

(SELECT EmpId FROM EmployeeSalary);
```

15. Write an SQL query to fetch the EmpIds that are present in EmployeeDetails but not in EmployeeSalary.

Using subquery-

```
SELECT EmpId FROM

EmployeeDetails

where EmpId Not IN

(SELECT EmpId FROM EmployeeSalary);
```

16. Write an SQL query to fetch the employee's full names and replace the space with '-'.

Using the 'Replace' function-

```
SELECT REPLACE(FullName, ' ', '-')
```

```
FROM EmployeeDetails;
```

17. Write an SQL query to fetch the position of a given character(s) in a field.
Using the ‘Instr’ function-

```
SELECT INSTR(FullName, 'Snow')

FROM EmployeeDetails;
```

18. Write an SQL query to display both the EmpId and ManagerId together.
Here we can use the CONCAT command.

```
SELECT CONCAT(EmpId, ManagerId) as NewId

FROM EmployeeDetails;
```

19. Write a query to fetch only the first name(string before space) from the FullName column of the EmployeeDetails table.
In this question, we are required to first fetch the location of the space character in the FullName field and then extract the first name out of the FullName field.

For finding the location we will use the LOCATE method in MySQL and CHARINDEX in SQL SERVER and for fetching the string before space, we will use the SUBSTRING OR MID method.

MySQL – using MID

```
SELECT MID(FullName, 1, LOCATE(' ',FullName))

FROM EmployeeDetails;
```

SQL Server – using SUBSTRING

```
SELECT SUBSTRING(FullName, 1, CHARINDEX(' ',FullName))

FROM EmployeeDetails;
```

20. Write an SQL query to uppercase the name of the employee and

lowercase the city values.
We can use SQL Upper and Lower functions to achieve the intended results.

```
SELECT UPPER(FullName), LOWER(City)

FROM EmployeeDetails;
```

21. Write an SQL query to find the count of the total occurrences of a particular character – ‘n’ in the FullName field.
Here, we can use the ‘Length’ function. We can subtract the total length of the FullName field from the length of the FullName after replacing the character – ‘n’.

```
SELECT FullName,

LENGTH(FullName) - LENGTH(REPLACE(FullName, 'n', ''))

FROM EmployeeDetails;
```

22. Write an SQL query to update the employee names by removing leading and trailing spaces.
Using the ‘Update’ command with the ‘LTRIM’ and ‘RTRIM’ functions.

```
UPDATE EmployeeDetails

SET FullName = LTRIM(RTRIM(FullName));
```

23. Fetch all the employees who are not working on any project.
This is one of the very basic interview questions in which the interviewer wants to see if the person knows about the commonly used – Is NULL operator.

```
SELECT EmpId

FROM EmployeeSalary

WHERE Project IS NULL;
```

24. Write an SQL query to fetch employee names having a salary greater than or equal to 5000 and less than or equal to 10000.
Here, we will use BETWEEN in the ‘where’ clause to return the EmpId of the

employees with salary satisfying the required criteria and then use it as a subquery to find the fullName of the employee from the EmployeeDetails table.

```
SELECT FullName

FROM EmployeeDetails

WHERE EmpId IN

(SELECT EmpId FROM EmployeeSalary

WHERE Salary BETWEEN 5000 AND 10000);
```

25. Write an SQL query to find the current date-time.

MySQL-

```
SELECT NOW();
```

SQL Server-

```
SELECT getdate();
```

Oracle-

```
SELECT SYSDATE FROM DUAL;
```

26. Write an SQL query to fetch all the Employee details from the EmployeeDetails table who joined in the Year 2020.

Using BETWEEN for the date range ’01-01-2020′ AND ’31-12-2020′-

```
SELECT * FROM EmployeeDetails

WHERE DateOfJoining BETWEEN '2020/01/01'

AND '2020/12/31';
```

Also, we can extract the year part from the joining date (using YEAR in MySQL)-

```
SELECT * FROM EmployeeDetails

WHERE YEAR(DateOfJoining) = '2020';
```

27. Write an SQL query to fetch all employee records from the EmployeeDetails table who have a salary record in the EmployeeSalary table.
Using 'Exists'-

```
SELECT * FROM EmployeeDetails E

WHERE EXISTS

(SELECT * FROM EmployeeSalary S

WHERE  E.EmpId = S.EmpId);
```

28. Write an SQL query to fetch the project-wise count of employees sorted by project's count in descending order.
The query has two requirements – first to fetch the project-wise count and then to sort the result by that count.

For project-wise count, we will be using the GROUP BY clause and for sorting, we will use the ORDER BY clause on the alias of the project count.

```
SELECT Project, count(EmpId) EmpProjectCount

FROM EmployeeSalary

GROUP BY Project

ORDER BY EmpProjectCount DESC;
```

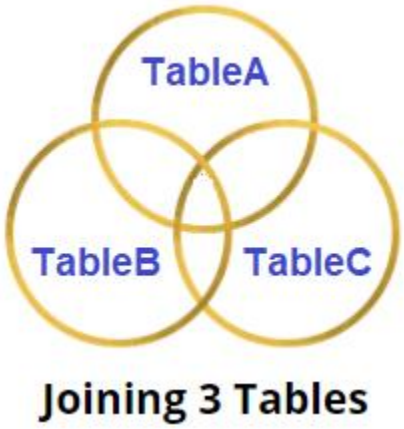
29. Write a query to fetch employee names and salary records. Display the employee details even if the salary record is not present for the employee.
This is again one of the very common interview questions in which the interviewer just wants to check the basic knowledge of SQL JOINS.

Here, we can use the left join with the EmployeeDetail table on the left side of the EmployeeSalary table.

```
SELECT E.FullName, S.Salary

FROM EmployeeDetails E

LEFT JOIN
```

```
EmployeeSalary S

ON E.EmpId = S.EmpId;
```

30. Write an SQL query to join 3 tables.
Considering 3 tables TableA, TableB, and TableC, we can use 2 joins clauses like below-




```
SELECT column1, column2

FROM TableA

JOIN TableB ON TableA.Column3 = TableB.Column3

JOIN TableC ON TableA.Column4 = TableC.Column4;
```

*For more questions on SQL Joins, you can also check our top **SQL Joins Interview Questions**.*

SQL Query Interview Questions for Experienced

Here is a list of some of the most frequently asked SQL query interview questions for experienced professionals. These questions cover SQL queries on advanced SQL JOIN concepts, fetching duplicate rows, odd and even rows, nth highest salary, etc.

31. Write an SQL query to fetch all the Employees who are also managers from the EmployeeDetails table.
Here, we have to use Self-Join as the requirement wants us to analyze the

EmployeeDetails table as two tables. We will use different aliases ‘E’ and ‘M’ for the same EmployeeDetails table.

```
SELECT DISTINCT E.FullName

FROM EmployeeDetails E

INNER JOIN EmployeeDetails M

ON E.EmpID = M.ManagerID;
```

32. Write an SQL query to fetch duplicate records from EmployeeDetails (without considering the primary key – EmpId).

In order to find duplicate records from the table, we can use GROUP BY on all the fields and then use the HAVING clause to return only those fields whose count is greater than 1 i.e. the rows having duplicate records.

```
SELECT FullName, ManagerId, DateOfJoining, City, COUNT(*)

FROM EmployeeDetails

GROUP BY FullName, ManagerId, DateOfJoining, City

HAVING COUNT(*) > 1;
```

33. Write an SQL query to remove duplicates from a table without using a temporary table.

Here, we can use delete with alias and inner join. We will check for the equality of all the matching records and then remove the row with a higher EmpId.

```
DELETE E1 FROM EmployeeDetails E1

INNER JOIN EmployeeDetails E2

WHERE E1.EmpId > E2.EmpId

AND E1.FullName = E2.FullName

AND E1.ManagerId = E2.ManagerId

AND E1.DateOfJoining = E2.DateOfJoining

AND E1.City = E2.City;
```

34. Write an SQL query to fetch only odd rows from the table.

In case we have an auto-increment field e.g. EmpId then we can simply use the below query-

```
SELECT * FROM EmployeeDetails

WHERE MOD (EmpId, 2) <> 0;
```

In case we don't have such a field then we can use the below queries.

Using Row_number in SQL server and checking that the remainder when divided by 2 is 1-

```
SELECT E.EmpId, E.Project, E.Salary

FROM (

   SELECT *, Row_Number() OVER(ORDER BY EmpId) AS RowNumber

   FROM EmployeeSalary

) E

WHERE E.RowNumber % 2 = 1;
```

Using a user-defined variable in MySQL-

```
SELECT *

FROM (

    SELECT *, @rowNumber := @rowNumber+ 1 rn

    FROM EmployeeSalary

    JOIN (SELECT @rowNumber:= 0) r

   ) t

WHERE rn % 2 = 1;
```

35. Write an SQL query to fetch only even rows from the table.

In case we have an auto-increment field e.g. EmpId then we can simply use the below query-

```
SELECT * FROM EmployeeDetails

WHERE MOD (EmpId, 2) = 0;
```

In case we don’t have such a field then we can use the below queries.

Using Row_number in SQL server and checking that the remainder, when divided by 2, is 1-

```
SELECT E.EmpId, E.Project, E.Salary

FROM (

   SELECT *, Row_Number() OVER(ORDER BY EmpId) AS RowNumber

   FROM EmployeeSalary

) E

WHERE E.RowNumber % 2 = 0;
```

Using a user-defined variable in MySQL-

```
SELECT *

FROM (

    SELECT *, @rowNumber := @rowNumber+ 1 rn

    FROM EmployeeSalary

    JOIN (SELECT @rowNumber:= 0) r

   ) t

WHERE rn % 2 = 0;
```

36. Write an SQL query to create a new table with data and structure copied from another table.

```
CREATE TABLE NewTable

SELECT * FROM EmployeeSalary;
```

37. Write an SQL query to create an empty table with the same structure as some other table.

Here, we can use the same query as above with the False ‘WHERE’ condition-

```
CREATE TABLE NewTable

SELECT * FROM EmployeeSalary where 1=0;
```

38. Write an SQL query to fetch top n records.

In MySQL using LIMIT-

```
SELECT *

FROM EmployeeSalary

ORDER BY Salary DESC LIMIT N;
```

In SQL server using TOP command-

```
SELECT TOP N *

FROM EmployeeSalary

ORDER BY Salary DESC;
```

39. Write an SQL query to find the nth highest salary from a table.

Using Top keyword (SQL Server)-

```
SELECT TOP 1 Salary

FROM (

    SELECT DISTINCT TOP N Salary

    FROM Employee

    ORDER BY Salary DESC

    )

ORDER BY Salary ASC;
```

Using limit clause(MySQL)-

```
SELECT Salary

FROM Employee

ORDER BY Salary DESC LIMIT N-1,1;
```

40. Write SQL query to find the 3rd highest salary from a table without using the TOP/limit keyword.

This is one of the most commonly asked interview questions. For this, we will use a correlated subquery.

In order to find the 3rd highest salary, we will find the salary value until the inner query returns a count of 2 rows having a salary greater than other distinct salaries.

```
SELECT Salary

FROM EmployeeSalary Emp1

WHERE 2 = (

        SELECT COUNT( DISTINCT ( Emp2.Salary ) )

        FROM EmployeeSalary Emp2

        WHERE Emp2.Salary > Emp1.Salary

    )
```

For the nth highest salary-

```
SELECT Salary

FROM EmployeeSalary Emp1

WHERE N-1 = (

        SELECT COUNT( DISTINCT ( Emp2.Salary ) )

        FROM EmployeeSalary Emp2

        WHERE Emp2.Salary > Emp1.Salary

    )
```

Scenario-based SQL Query Interview Questions

Let's see some interview questions based on different scenarios. The questions are of varying difficulty levels and the goal is to prepare you for different real-time scenario-based questions.

41. Consider a `SalesData` with columns `SaleID`, `ProductID`, `RegionID`, `SaleAmount`. Write a query to find the total sales amount for each product in each region.

The below query sums up `SaleAmount` for each combination of `ProductID` and `RegionID`, giving an insight into the total sales per product per region.

```
SELECT ProductID, RegionID, SUM(SaleAmount) AS TotalSales

FROM SalesData

GROUP BY ProductID, RegionID;
```

42. Write a query to find employees who earn more than their managers.

Here, we will write a query that joins the `EmployeeDetails` table with itself to compare the salaries of employees with their respective managers.

```
SELECT E.Name AS EmployeeName,

M.Name AS ManagerName,

E.Salary AS EmployeeSalary,

M.Salary AS ManagerSalary

FROM EmployeeDetails E JOIN EmployeeDetails M

ON E.ManagerID = M.EmployeeID

WHERE E.Salary > M.Salary;
```

43. Consider a `BookCheckout` table with columns – `CheckoutID`, `MemberID`, `BookID`, `CheckoutDate`, `ReturnDate`. Write an SQL query to find the number of books checked out by each member.

```
SELECT MemberID, COUNT(*) AS NumberOfBooksCheckedOut
```

```
FROM BookCheckout

GROUP BY MemberID;
```

44. Consider a `StudentGrades` table with columns – `StudentID`, `CourseID`, `Grade`. Write a query to find students who have scored an 'A' in more than three courses.

Here we will write an SQL query that filters students who have received an 'A' grade and groups them by `StudentID`, counting the number of 'A' grades per student.

```
SELECT StudentID FROM StudentGrades

WHERE Grade = 'A'

GROUP BY StudentID

HAVING COUNT(*) > 3;
```

45. Consider a table `OrderDetails` with columns – `OrderID`, `CustomerID`, `ProductID`, `OrderDate`, `Quantity`, `Price`. Write a query to find the average order value for each customer.

The below query calculates the average order value (quantity multiplied by price) for each customer.

```
SELECT CustomerID, AVG(Quantity * Price) AS AvgOrderValue

FROM OrderDetails

GROUP BY CustomerID;
```

46. Consider a table `PatientVisits` with Columns `VisitID`, `PatientID`, `DoctorID`, `VisitDate`, `Diagnosis`. Write a query to find the latest visit date for each patient.

```
SELECT PatientID, MAX(VisitDate) AS LatestVisitDate

FROM PatientVisits

GROUP BY PatientID;
```

47. For a table `FlightBookings` with columns – `BookingID`, `FlightID`, `PassengerID`, `BookingDate`, `TravelDate`, `Class`, write a query to count the number of bookings for each flight class.

Here, we will write an SQL query that groups the bookings by `Class` and counts the number of bookings in each class.

```
SELECT Class, COUNT(*) AS NumberOfBookings

FROM FlightBookings

GROUP BY Class;
```

48. Consider a table `FoodOrders` with columns – `OrderID`, `TableID`, `MenuItemID`, `OrderTime`, `Quantity`. Write a query to find the most ordered menu item.

For the desired output, we will group the orders by `MenuItemID` and then sort the results by the count in descending order, fetching the top result.

```
SELECT MenuItemID

FROM FoodOrders

GROUP BY MenuItemID

ORDER BY COUNT(*) DESC

LIMIT 1;
```

49. Consider a table `Transactions` with columns – `TransactionID`, `CustomerID`, `ProductID`, `TransactionDate`, `Amount`. Write a query to find the total transaction amount for each month.

The below query sums the `Amount` for each month, giving a monthly total transaction amount.

```
SELECT MONTH(TransactionDate) AS Month,

SUM(Amount) AS TotalAmount

FROM Transactions

GROUP BY MONTH(TransactionDate);
```

50. Consider a table `EmployeeAttendance` with columns

– `AttendanceID`, `EmployeeID`, `Date`, `Status`. **Write a query to find employees with more than 5 absences in a month.**
This query filters the records for absent status, groups them by `EmployeeID` and month, and counts absences, filtering for more than 5 absences.

```
SELECT EmployeeID,

MONTH(Date) AS Month,

COUNT(*) AS Absences

FROM EmployeeAttendance

WHERE Status = 'Absent'

GROUP BY EmployeeID, MONTH(Date)

HAVING COUNT(*) > 5;
```

This concludes our post on frequently asked **SQL query interview questions and answers**. I hope you practice these questions and ace your database interviews.