[SQL query to find third highest salary in company](http://stackoverflow.com/questions/20690629/sql-query-to-find-third-highest-salary-in-company)

select Max(salary)

from employees

where Salary not in

(select Max(salary)

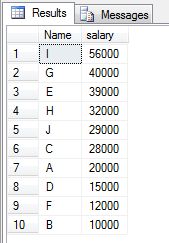
from employees

where Salary not in

(select Max(salary)

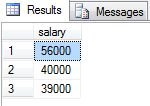
from employees));

***Another way***

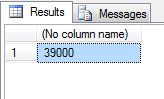
SELECT  Name,salary  FROM employee ORDER BY salary desc  
  


**Note**: I am assuming that we want to find the 3rd highest salary.  
  
  
  
**Explanation**: In the preceding query my motive is that first I will get the highest 3 salaries and then get the minimum salary from those 3 salaries.  
  
There are the following 2 parts of the preceding query:

1. Inner Query: Get the highest 3 salaries
2. Outer Query: Get the minimum salary from those 3 salaries

1. In the Inner Query I have used the "DISTINCT", "TOP", "ORDER BY" and "DESC" keywords, they mean:  
  
**DISTINCT**: for removing the duplicity.   
  
**TOP**: to get the number of upper rows from the set of records like here 3 is used as number.  
  
**ORDER BY**: to ordering the set of rows.  
  
**DESC**: used with "ORDER BY" to get the data in decreasing order.  
  
So by the inner query we will get output like this:  
  
  
  
2. Now to find the minimum salary from the inner query. To do that I will write the outer query using the "MIN" keyword and aliase the set by "as a" to get the final output like this:  
  
SELECT MIN( salary)  
FROM (  
-- INNER Query  
 ) as a

**MIN**: to get the minimum record from the set.  
  
**as a:** for the aliasing of the set of records like here the result of the inner query is aliasing "as a".  
  
Final **Output:** The output will be 39000.



## SQL Aggregate Functions

SQL aggregate functions return a single value, calculated from values in a column.

Useful aggregate functions:

* AVG() - Returns the average value
* COUNT() - Returns the number of rows
* FIRST() - Returns the first value
* LAST() - Returns the last value
* MAX() - Returns the largest value
* MIN() - Returns the smallest value
* SUM() - Returns the sum

## SQL Scalar functions

SQL scalar functions return a single value, based on the input value.

Useful scalar functions:

* UCASE() - Converts a field to upper case
* LCASE() - Converts a field to lower case
* MID() - Extract characters from a text field
* LEN() - Returns the length of a text field
* ROUND() - Rounds a numeric field to the number of decimals specified
* NOW() - Returns the current system date and time
* FORMAT() - Formats how a field is to be displayed

**Tip:** The aggregate functions and the scalar functions will be explained in details in the next chapters.

### [what is stored procedure in Sql server | what are the advantages of using stored procedures in sql server](http://www.aspdotnet-suresh.com/2011/07/what-is-stored-procedure-in-sql-server.html)

**Description:**

A stored procedure is a group of sql statements that has been created and stored in the database. Stored procedure will accept input parameters so that a single procedure can be used over the network by several clients using different input data. Stored procedure will reduce network traffic and increase the performance.

**Advantages of using stored procedures**

 a) Stored procedure allows modular programming.

You can create the procedure once, store it in the database, and call it any number of times in your program.

b) Stored Procedure allows faster execution.

If the operation requires a large amount of SQL code is performed repetitively, stored procedures can be faster. They are parsed and optimized when they are first executed, and a compiled version of the stored procedure remains in memory cache for later use. This means the stored procedure does not need to be reparsed and re optimized with each use resulting in much faster execution times.

 c) Stored Procedure can reduce network traffic.

An operation requiring hundreds of lines of Transact-SQL code can be performed through a single statement that executes the code in a procedure, rather than by sending hundreds of lines of code over the network.

d) Stored procedures provide better security to your data

Users can be granted permission to execute a stored procedure even if they do not have permission to execute the procedure's statements directly.

In SQL we are having different types of stored procedures are there

a)    System Stored Procedures

b)    User Defined Stored procedures

c)    Extended Stored Procedures

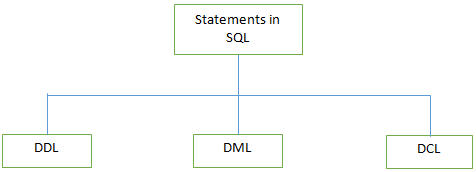
# SQL JOIN

## Different types of JOINs

* (INNER) JOIN: Select records that have matching values in both tables.
* LEFT (OUTER) JOIN: Select records from the first (left-most) table with matching right table records.
* RIGHT (OUTER) JOIN: Select records from the second (right-most) table with matching left table records.
* FULL (OUTER) JOIN: Selects all records that match either left or right table records.



**Question #3) What are different types of statements supported by SQL?**

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2016/07/statements-supported-by-SQL.jpg)

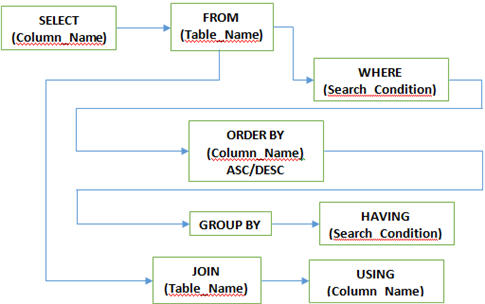
***There are 3 types of SQL statements***

**1) DDL (Data Definition Language):** It is used to define the database structure such as tables. It includes three statements such as **Create, Alter, and Drop.**

**2) DML (Data Manipulation Language):** These statements are used to manipulate the data in records. Commonly used DML statements are **Insert, Update, and Delete.**

**3) DCL (Data Control Language):**These statements are used to set privileges such as **Grant and Revoke** database access permission to the specific user**.**

**Question #5) What are different Clauses used in SQL?**

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2016/07/Clauses-used-in-SQL.jpg)

* **WHERE Clause:**This clause is used to define the condition, extract and display only those records which fulfill the given condition

***Syntax:****SELECT column\_name(s)*  
*FROM table\_name*  
*WHERE condition;*

* **GROUP BY Clause:** It is used with SELECT statement to group the result of the executed query using the value specified in it. It matches the value with the column name in tables and groups the end result accordingly.

***Syntax:****SELECT column\_name(s)*  
*FROM table\_name*  
*GROUP BY column\_name;*

* **HAVING clause:**This clause is used in association with GROUP BY clause. It is applied to the each group of result or the entire result as single group and much similar as WHERE clause, the only difference is you cannot use it without GROUP BY clause

***Syntax:****SELECT column\_name(s)*  
*FROM table\_name*  
*GROUP BY column\_name*  
*HAVING condition;*

* **ORDER BY clause:**This clause is to define the order of the query output either in ascending (ASC) or in descending (DESC) order. Ascending (ASC) is the default one but descending (DESC) is set explicitly.

***Syntax:****SELECT column\_name(s)*  
*FROM table\_name*  
*WHERE condition*  
*ORDER BY column\_name ASC|DESC;*

* **USING clause:**USING clause comes in use while working with SQL Joins. It is used to check equality based on columns when tables are joined. It can be used instead ON clause in Joins.

***Syntax:****SELECT column\_name(s)*  
*FROM table\_name*  
*JOIN table\_name*  
*USING (column\_name);*

* **COMMIT**: It is used to save all changes made through the transaction
* **ROLLBACK**: It is used to roll back the transaction such as all changes made by the

**Question #22) What is the Cartesian product of table?**

The output of Cross Join is called as a Cartesian product. It returns rows combining each row from the first table with each row of the second table. For Example, if we join two tables having 15 and 20 columns the Cartesian product of two tables will be 15×20=300 Rows.

**Question #23) What do you mean by Subquery?**

Query within another query is called as Subquery. A subquery is called inner query which returns output that is to be used by another query.

**Question #26) What is the difference between DELETE and TRUNCATE?**

* The basic difference in both is DELETE is DML command and TRUNCATE is DDL
* DELETE is used to delete a specific row from the table whereas TRUNCATE is used to remove all rows from the table
* We can use DELETE with WHERE clause but cannot use TRUNCATE with it

**Question #27) What is the difference between DROP and TRUNCATE?**

TRUNCATE removes all rows from the table which cannot be retrieved back, DROP removes the entire table from the database and it cannot be retrieved back.

**Question #28) How to write a query to show the details of a student from Students table whose  
name starts with K?**

*SELECT \* FROM Student WHERE Student\_Name like ‘%K’;*

**Q#44. What is the syntax to add a record to a table?**  
**Ans.** To add a record in a table INSERT syntax is used.

Ex: INSERT into table\_name VALUES (value1, value2..);

**Q#45. How do you add a column to a table?**  
**Ans.** To add another column in the table following command has been used.

ALTER TABLE table\_name ADD (column\_name);

**Q#46. Define SQL Delete statement.**  
**Ans.** Delete is used to delete a row or rows from a table based on the specified condition.  
The basic syntax is as follows:

DELETE FROM table\_name

WHERE <Condition>

**Q#47. Define COMMIT?**  
**Ans.** COMMIT saves all changes made by DML statements.

**Q#65. Difference between TRUNCATE, DELETE and DROP commands?**  
**Ans.** DELETE removes some or all rows from a table based on the condition. It can be rolled back.

TRUNCATE removes ALL rows from a table by de-allocating the memory pages. The operation cannot be rolled back

DROP command removes a table from the database completely.

**Q#88. What is the difference between Having clause and Where clause?**  
**Ans.** Both specify a search condition but Having clause is used only with the SELECT statement and typically used with GROUP BY clause.  
If GROUP BY clause is not used then Having behaves like WHERE clause only.

**Example1: Suppose we want to fetch records from two tables  emp, dept where deptno of employee is equal to dept no of dept.**

SELECT \* FROM EMP E INNER JOIN DEPT  ON D WHERE E.DEPTNO=D.DEPTNO;

**Question 3: Write SQL Query to display the current date.**

SELECT GetDate();

**Question 4: Write an SQL Query to check whether date passed to Query is the date of given format or not**

SELECT  ISDATE('1/08/13') AS "MM/DD/YY";

**Question 6: Write an SQL Query find number of employees according to gender  whose DOB is between 01/01/1960 to 31/12/1975.**

SELECT COUNT(\*), sex from Employees WHERE DOB BETWEEN '01/01/1960' AND '31/12/1975' GROUP BY sex;

**Question 7: Write an SQL Query to find an employee whose Salary is equal or greater than 10000**.

SELECT EmpName FROM Employees WHERE Salary>=10000;

**Question 8: Write an SQL Query to find name of employee whose name Start with ‘M’**

SELECT \* FROM Employees WHERE EmpName like 'M%';

**Question 9: find all Employee records containing the word "Joe", regardless of whether it was stored as JOE, Joe, or joe.**

SELECT \* from Employees WHERE UPPER(EmpName) like '%JOE%';

**Question 10: Write an SQL Query to find  the year from date.**

SELECT YEAR(GETDATE()) as "Year";

**Question 11: Write SQL Query to find duplicate rows in a database? and then write SQL query to delete them?**

SELECT \* FROM emp a WHERE rowid = (SELECT MAX(rowid) FROM EMP b WHERE a.empno=b.empno)

DELETE FROM emp a WHERE rowid != (SELECT MAX(rowid) FROM emp b WHERE a.empno=b.empno);

Hello friends! in this post we will see some of the most commonly asked SQL queries in interviews. The questions will start from very basic questions and then move to more complex problems. Consider the below two tables for most of the questions asked here.

**Table - EmployeeDetails**

|  |  |  |  |
| --- | --- | --- | --- |
| **EmpId** | **FullName** | **ManagerId** | **DateOfJoining** |
| 121 | John Snow | 321 | 01/31/2014 |
| 321 | Walter White | 986 | 01/30/2015 |
| 421 | Kuldeep Rana | 876 | 27/11/2016 |

**Table - EmployeeSalary**

|  |  |  |
| --- | --- | --- |
| **EmpId** | **Project** | **Salary** |
| 121 | P1 | 8000 |
| 321 | P2 | 1000 |
| 421 | P1 | 12000 |

## SQL Query Interview Questions with Answers

**Ques.1. Write a SQL query to fetch the count of employees working in project 'P1'.**  
Ans. Here, we would be using aggregate function count() with the SQL where clause-

**SELECT** **COUNT**(\*) **FROM** EmployeeSalary **WHERE** Project = 'P1';

**Ques.2. Write a SQL query to fetch employee names having salary greater than or equal to 5000 and less than or equal 10000.**  
Ans. Here, we will use BETWEEN in the 'where' clause to return the empId of the employees with salary satifying the required criteria and then use it as subquery to find the fullName of the employee form EmployeeDetails table.

**SELECT** FullName

**FROM** EmployeeDetails

**WHERE** EmpId **IN**

(**SELECT** EmpId **FROM** EmpolyeeSalary

**WHERE** Salary **BETWEEN** 5000 **AND** 10000);

**Ques.3. Write a SQL query to fetch project-wise count of employees sorted by project's count in descending order.**  
Ans. 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 GROUPBY clause and for sorting, we will use ORDER BY clause on the alias of the project-count.

**SELECT** Project, **count**(EmpId) EmpProjectCount

**FROM** EmployeeSalary

**GROUP** **BY** Project

**ORDER** **BY** EmpProjectCount **DESC**;

**Ques.4. Write a query to fetch only the first name(string before space) from the FullName column of EmployeeDetails table.**  
Ans. 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 LOCATE method in mySQL and CHARINDEX in SQL SERVER and for fetching the string before space, we will use SUBSTRING OR MID method.

**mySQL- Using MID**

**SELECT** MID(FullName, 0, LOCATE(' ',FullName)) **FROM** EmployeeDetails;

**SQL Server-Using SUBSTRING**

**SELECT** SUBSTRING(FullName, 0, CHARINDEX(' ',FullName)) **FROM** EmployeeDetails;

**Also, we can use LEFT which returns the left part of a string till specified number of characters.**

**SELECT** **LEFT**(FullName, CHARINDEX(' ',FullName) - 1) **FROM** EmployeeDetails;

**Ques.5. Write a query to fetch employee names and salary records. Return employee details even if the salary record is not present for the employee.**  
Ans. Here, we can use left join with EmployeeDetail table on the left side.

**SELECT** E.FullName, S.Salary

**FROM** EmployeeDetails E **LEFT** **JOIN** EmployeeSalary S

**ON** E.EmpId = S.EmpId;

**Ques.6. Write a SQL query to fetch all the Employees who are also managers from EmployeeDetails table.**  
Ans. Here, we have to use Self-Join as the requirement wants us to analyze the EmployeeDetails table as two different tables, each for Employee and manager records.

**SELECT DISTINCT** E.FullName

**FROM** EmpDetails E

**INNER** **JOIN** EmpDetails M

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

**Ques.7. Write a SQL query to fetch all employee records from EmployeeDetails table who have a salary record in EmployeeSalary table.**  
Ans. Using 'Exists'-

**SELECT** \* **FROM** EmployeeDetails E

**WHERE** **EXISTS**

(**SELECT** \* **FROM** EmployeeSalary S **WHERE** E.EmpId = S.EmpId);

**Ques.8. Write a SQL query to fetch duplicate records from a table.**  
Ans. In order to find duplicate records from table we can use GROUP BY on all the fields and then use HAVING clause to return only those fields whose count is greater than 1 i.e. the rows having duplicate records.

**SELECT** EmpId, Project, Salary, **COUNT**(\*)

**FROM** EmployeeSalary

**GROUP** **BY** EmpId, Project, Salary

**HAVING** **COUNT**(\*) > 1;

**Ques.9. Write a SQL query to remove duplicates from a table without using temporary table.**  
Ans. Using Group By and Having clause-

**DELETE** **FROM** EmployeeSalary

**WHERE** EmpId **IN** (

**SELECT** EmpId

**FROM** EmployeeSalary

**GROUP** **BY** Project, Salary

**HAVING** **COUNT**(\*) > 1));

Using rowId in Oracle-

**DELETE** **FROM** EmployeeSalary

**WHERE** rowid **NOT** **IN**

(**SELECT** **MAX**(rowid) **FROM** EmployeeSalary **GROUP** **BY** EmpId);

**Ques.10. Write a SQL query to fetch only odd rows from table.**  
Ans. This can be achieved by using Row\_number in SQL server-

**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

**Ques.11. Write a SQL query to fetch only even rows from table.**  
Ans. Using the same Row\_Number() and checking that the remainder when divided by 2 is 0-

**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

**Ques.12. Write a SQL query to create a new table with data and structure copied from another table.**  
Ans. Using SELECT INTO command-

**SELECT** \* **INTO** newTable **FROM** EmployeeDetails;

**Ques.13. Write a SQL query to create an empty table with same structure as some other table.**  
Ans. Using SELECT INTO command with False 'WHERE' condition-

**SELECT** \* **INTO** newTable **FROM** EmployeeDetails **WHERE** 1 = 0;

This can also done using mySQL 'Like' command with CREATE statement-

**CREATE** **TABLE** newTable **LIKE** EmployeeDetails;

**Ques.14. Write a SQL query to fetch common records between two tables.**  
Ans. Using INTERSECT-

**SELECT** \* **FROM** EmployeeSalary

**INTERSECT**

**SELECT** \* **FROM** ManagerSalary

**Ques.15. Write a SQL query to fetch records that are present in one table but not in another table.**  
Ans. Using MINUS-

**SELECT** \* **FROM** EmployeeSalary

**MINUS**

**SELECT** \* **FROM** ManagerSalary

**Ques.16. Write a SQL query to find current date-time.**  
Ans. mySQL-

**SELECT** NOW();

SQL Server-

**SELECT** getdate();

Oracle-

**SELECT** SYSDATE **FROM** DUAL;

**Ques.17. Write a SQL query to fetch all the Employees from EmployeeDetails table who joined in Year 2016.**  
Ans. Using BETWEEN for the date range '01-01-2016' AND '31-12-2016'-

**SELECT** \* **FROM** EmployeeSalary

**WHERE** DateOfJoining **BETWEEN** '01-01-2016' **AND** date '31-12-2016';

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

**SELECT** \* **FROM** EmployeeSalary

**WHERE** **YEAR**(DateOfJoining) = '2016';

**Ques.18. Write a SQL query to fetch top n records?**  
Ans. 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**

In Oracle using ROWNUM-

**SELECT** \* **FROM** (**SELECT** \* **FROM** EmployeeSalary **ORDER** **BY** Salary **DESC**)

**WHERE** ROWNUM <= 3;

**Ques.19. Write SQL query to find the nth highest salary from table.**  
Ans. 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;

**Ques.20. Write SQL query to find the 3rd highest salary from table without using TOP/limit keyword.**  
Ans. The below SQL query make use of correlated subquery wherein in order to find the 3rd highest salary the inner query will return the count of till we find that there are two rows that 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 nth highest salary-

**SELECT** Salary

**FROM** EmployeeSalary Emp1

**WHERE** N-1 = (

**SELECT** **COUNT**( **DISTINCT** ( Emp2.Salary ) )

**FROM** EmployeeSalary Emp2

**WHERE** Emp2.Salary > Emp1.Salary

)