SQL Query Interview Questions

In this article, you will learn many simple and complex SQL queries asked in IT interviews. Let's take two tables which help in solving various queries. The name of the first table is **Student,** and the name of the second table is **Subject.**

The Student table consists of **Student_ID, Stu_Name, Stu_Subject_ID, Stu_Marks, and Stu_Age** columns, and the **Subject** table consists of **Subject_ID and Subject_Name** columns.

Student Table:

101	Akhil	BCA101	85	20
102	Balram	BCA104	78	19
103	Bheem	BCA102	80	22
104	Chetan	BCA103	95	20
105	Diksha	BCA104	99	20
106	Raman	BCA105	88	19
107	Sheetal	BCA103	98	22

Column Table:

BCA101	C

BCA102	C++
BCA103	Principle of Management
BCA104	Core Java
BCA105	Math
BCA106	Android

Query 1: Write a query to create the table in Structured Query Language.

Sol:

Syntax to Create a Table in SQL:

```
CREATE TABLE table_name
(
column_Name1 data type (size of the column),
column_Name2 data type (size of the column),
column_Name3 data type (size of the column),
...
column_NameN data type (size of the column)
);
```

We can create a table using **Create Table** keyword. This keyword creates only one table at a time.

Examples:

Example 1:

The following example creates the Student table:

```
CREATE TABLE Student
(
Student_ID int,
Stu_Name varchar (25),
Stu_Subject_ID varchar (10),
Stu_Marks int,
Stu_Age int
);
```

Example 2:

The following example creates the Subject table:

```
CREATE TABLE Subject
(
Subject_ID varchar (10),
Subject_Name varchar (30),
);
```

Query 2: Write a query to insert the data into the table.

Sol:

Syntax to insert data into a table:

```
INSERT INTO Table_Name VALUES (value_1, value_2, value_3, ...., value_N);
```

We can easily insert the record using the INSERT statement in SQL.

Examples:

Example 1:

The following queries insert the data of students into Student table:

```
INSERT INTO Student VALUES (101, Akhil, BCA101, 85, 20);
INSERT INTO Student VALUES (102, Balram, BCA104, 78, 19);
INSERT INTO Student VALUES (103, Bheem, BCA102, 80, 22);
INSERT INTO Student VALUES (104, Chetan, BCA103, 95, 20);
INSERT INTO Student VALUES (105, Diksha, BCA104, 99, 20);
INSERT INTO Student VALUES (106, Raman, BCA105, 88, 19);
INSERT INTO Student VALUES (107, Sheetal, BCA103, 98, 22);
```

Example 2:

The following query inserts Subject_ID and Subject_Name into the Subject table:

```
INSERT INTO Subject VALUES (BCA101, C);
INSERT INTO Subject VALUES (BCA102, C++);
INSERT INTO Subject VALUES (BCA103, Principle of Management);

INSERT INTO Subject VALUES (BCA104, Core Java);
INSERT INTO Subject VALUES (BCA105, Math);
INSERT INTO Subject VALUES (BCA106, Android);
```

Query 3: Write a query to view the specific record of the table by using the WHERE clause.

Sol:

Syntax to access specific records from the table:

```
SELECT * FROM Table_Name WHERE condition;
```

Examples:

Example 1:

The following query shows all the rows of those Students whose age is 20:

```
SELECT * FROM Student WHERE Stu_Age = 20;
```

The WHERE clause in this query shows only those rows which satisfy the specified condition.

Output:

101	Akhil	BCA101	85	20
104	Chetan	BCA103	95	20
105	Diksha	BCA104	99	20

Example 2: The following query shows the Subject_Name of those subjects whose Subject_ID is BCA103 and BCA106:

```
SELECT * FROM Student WHERE Subject_ID = 'BCA103' and Subject_ID = 'BCA106' ;
```

The WHERE clause in this query shows only those rows which satisfy the specified condition.

Output:

BCA103	Principle of Management
BCA106	Android

Query 4: Write a query in SQL to find the minimum and maximum number from the integer column:

Sol:

Syntax to find the maximum and minimum number from the column:

```
SELECT MAX(Column_Name), MIN(Column_Name) FROM Table_Name;
```

We can easily find the maximum and minimum values of any integer column using the MAX and MIN aggregate functions.

Example:

The following query shows the maximum and minimum marks of the Stu_Marks column from the Student table:

```
SELECT MAX(Stu_Marks), MIN(Stu_Marks) FROM Student;
```

Query 5: Write a query to access the first record from the SQL table?

Sol:

Syntax to find the first record from the table:

```
SELECT * FROM Table_Name WHERE Rownum = 1;
```

We can easily find the first row of any table by assigning 1 to the Rownum keyword in the WHERE clause of the SELECT statement.

Example:

The following query shows the first row of the student table in the output:

```
SELECT * FROM Student WHERE Rownum = 1;
```

Output:

101	Akhil	BCA101	85	20

Query 6: Write a query to access the last record from the table?

Sol:

Syntax to find the first record from the table:

```
SELECT * FROM Table_Name WHERE Rowid = SELECT MAX(Rowid) from Table_Name;
```

We can easily find the last row of any table by using the above syntax.

Example:

The following query shows the last row of the student table in the output:

```
SELECT * FROM Student WHERE Rowid = SELECT MAX(Rowid) from Student;
```

Output:

107	Sheetal	BCA103	98	22

Query 7: Write a query to access the first Nth rows from the table?

Sol:

Syntax to find the first Nth records from the table:

```
SELECT * FROM Table_Name WHERE Rownum < = N ;
```

We can easily retrieve the first five rows of any table by using the Rownum keyword. We have to use the 'Less than equals to' comparison operator for this operation.

Here, N defines the number of rows to be shown in the output.

Example:

The following query shows the first five rows of the student table in the output:

```
SELECT * FROM Student WHERE Rownum < = 5;
```

Output:

101	Akhil	BCA101	85	20
102	Balram	BCA104	78	19
103	Bheem	BCA102	80	22

104	Chetan	BCA103	95	20
105	Diksha	BCA104	99	20

Query 8: Write a query to access the last Nth rows from the SQL table?

Sol:

Syntax to find the last Nth records from the table:

```
SELECT * FROM (SELECT * FROM Table_Name order by Rowid DESC) WHERE Rownum < = N ;
```

We can easily retrieve the first five rows of any table by using the Rownum keyword.

Example:

The following query shows the last four rows of the Subject table:

```
SELECT * FROM (SELECT * FROM Subject order by Rowid DESC) WHERE Rownum < = 4 ;
```

Output:

BCA103	Principle of Management
BCA104	Core Java
BCA105	Math
BCA106	Android

Query 9: Write a query in SQL to retrieve only even rows from the table?

Sol:

Syntax to find the Even rows from the table:

```
SELECT * FROM Table_Name WHERE MOD (Rowid,2) = 0 ;
```

We can easily retrieve the even rows from the table by using the MOD function in the WHERE clause of the SELECT statement.

Example:

The following query shows even rows of student table in the result:

```
SELECT * FROM Student WHERE MOD (Rowid,2) = 0 ;
```

Output:

102	Balram	BCA104	78	19
104	Chetan	BCA103	95	20
106	Raman	BCA105	88	19

Query 10: Write a query in SQL to retrieve only an odd number of rows from the table?

Sol:

Syntax to find the Odd number of rows from the table:

```
SELECT * FROM Table_Name WHERE MOD (Rowid,2) = 1 ;
```

We can easily retrieve the odd rows from the table by using the MOD function in the WHERE clause of the SELECT statement.

Example:

The following query shows odd rows of Student table in the result:

```
SELECT * FROM Student WHERE MOD (Rowid,2) = 1 ;
```

Output:

101	Akhil	BCA101	85	20
103	Bheem	BCA102	80	22
105	Diksha	BCA104	99	20
107	Sheetal	BCA103	98	22

Query 11: Write a query in SQL to create a new table with the same data and structure as an existing table.

Sol:

Syntax:

```
CREATE TABLE New_Table_Name SELECT * FROM Existing_Table_Name;
```

Example: The following query creates Student_Marks table from the existing Student table:

```
CREATE TABLE Student_Marks SELECT * FROM Student;
```

Query 12: Write a Query to find the Nth highest value of an integer column from the table.

Sol:

Syntax:

```
SELECT TOP 1 Column_Name
FROM (
SELECT DISTINCT TOP N Column_Name
FROM Table_Name
ORDER BY Column_Name DESC
)
ORDER BY Column_Name ASC;
```

Example:

The following query shows the 3rd highest marks from the Student table:

```
SELECT TOP 1 Stu_Marks
FROM (
SELECT DISTINCT TOP N Stu_Marks
FROM Student
ORDER BY Stu_Marks DESC
)
ORDER BY Stu_Marks ASC;
```

Query 13: Write a query in SQL to find the second-highest value of an integer column from the table?

Sol:

Syntax to find the second highest value of the integer column:

```
Select MAX(Column_Name) from Table_Name
where Column_Name NOT IN (Select MAX(Column_Name) from Table_Name);
```

Example:

The following query shows the second-highest marks from the student table:

```
Select MAX(Stu_Marks) from Student
where Stu_Marks NOT IN (Select MAX(Stu_Marks) from Student);
```

Query 14: Write a query in Structured Query Language to view the current date and time.

Sol:

```
SELECT GETDATE();
```

Query 15: Write a query in SQL to show the record of the three highest values of an integer column from the table.

Sol:

Syntax:

```
SELECT (Column_Name) FROM (SELECT DISTINCT Column_Name from Table_Name ORDER BY Column_Name DESC) WHERE ROWNUM<=3;
```

Example:

The following query shows the record of the three highest marks from the student table:

```
SELECT (Stu_Marks) FROM (SELECT DISTINCT Stu_Marks from Student ORDER BY Stu_Marks DESC) WHERE ROWNUM<=3;
```

Output:

105	Diksha	BCA104	99	20
107	Sheetal	BCA103	98	22
104	Chetan	BCA103	95	20

Query 16: Write an SQL query to fetch the Stu_Name and Stu_Marks of those students whose age is 20.

Sol:

For this operation, you have to use the WHERE clause in the SELECT statement.

```
SELECT Stu_Name, Stu_Marks FROM Student WHERE Stu_Age = 20;
```

Output:

Akhil	85
Chetan	95
Diksha	99

Query 17: Write a query to show the maximum marks of each subject.

Sol:

For this operation, you need to use the MAX function with the GROUP BY statement.

```
Select Student_ID, Stu_Subject_ID, MAX(Stu_Marks) from Student group by Stu_Subject_ID;
```

Output:

101	BCA101	85

105	BCA104	99
103	BCA102	80
107	BCA103	98
106	BCA105	88

Query 18: Write a query to show all the record of those students whose Marks is greater than 82 and age is 22

Sol:

Here, you have to use the AND operator between the two conditions in the WHERE clause. The AND operator returns those records which match the specified conditions.

```
SELECT * FROM Student WHERE Stu_Marks > 82 and Stu_Age = 22;
```

Output:

107	Sheetal	BCA103	98	22

Query 19: Write a query to show the record of those students whose name begins with the 'm' character.

Sol:

Here, you have to use the LIKE operator, which matches the given pattern in the table.

```
SELECT * FROM Student WHERE Stu_Name LIKE '%m';
```

Output:

102	Balram	BCA104	78	19
103	Bheem	BCA102	80	22

Query 20: Write a query to show all Subject_ID along with the number of students in there.

Sol: The following query uses the GROUP BY statement with the COUNT function, which returns the number of students in each subject.

```
SELECT Stu_Subject_ID COUNT(Stu_Subject_ID) as 'Number of Students' FROM Student GROUP BY Stu_Subject_ID ;
```

Output:

BCA101	1
BCA104	2
BCA102	1
BCA103	2
BCA105	1

Query 21: Write a query in SQL to fetch the values of the Stu_Name column from the Student table in the upper case.

Sol:

The following query uses the **UPPER** function with that column name whose values are to be shown in upper case:

1. **SELECT** UPPER(Stu_Name) **from** Student;

Query 22: Write an SQL query to show the unique values of Stu_Age from the student table:

Sol:

The following query uses the SQL **DISTINCT** function with the **Stu_Age** column:

1. **SELECT DISTINCT**(Stu_Age) **from** Student;

Query 23: Write a Query in SQL to show the first N characters of the string column from the Student table.

Sol:

Syntax:

1. **SELECT** SUBSTRING(Column_Name, 1, N) **from** Table_Name;

This syntax uses the SUBSTRING function, which shows the specific characters of the string.

Example:

The following query shows the first two characters of Stu_Name from the Student table:

1. **SELECT** SUBSTRING(Stu_Name, 1, 2) **from** Student;

Query 24: Write a query in structured query language to view all student details from the Student table order by Stu_Name Descending.

Sol:

Here, we have to use the ORDER BY clause, which shows the student details in the descending order of Stu_Name:

```
SELECT * FROM Student ORDER BY Stu_Name DESC;
```

Output:

107	Sheetal	BCA103	98	22
106	Raman	BCA105	88	19
105	Diksha	BCA104	99	20
104	Chetan	BCA103	95	20
103	Bheem	BCA102	80	22
102	Balram	BCA104	78	19
101	Akhil	BCA101	85	20

Query 25: Write a query to show the values from one table that does not exist in another table in the same database.

Sol:

Syntax:

```
SELECT * FROM Table_Name1 MINUS SELECT * FROM Table_Name2;
```

This syntax uses the SQL MINUS operator, which shows the values of Table1 that does not exist in Table2.

Example: Let's take another table, **Student2,** consisting of 3 columns Bus_ID, Stu_Name, and Stu_Address.

1	Ramesh	BCA101
6	Chetan	BCA103
5	Akhil	BCA101
4	Bhanu	BCA103
3	Balram	BCA104
2	Ram	BCA105

The following query shows only those rows of Stu_Name and Stu_Subject_ID of student table which does not exist in Student2 table:

Output:

```
SELECT Stu_Name, Stu_Subject_ID from Student
MINUS
SELECT Stu_Name, Stu_Subject_ID from Student2;
```

Output:

Bheem	BCA102

Diksha	BCA104
Raman	BCA105
Sheetal	BCA103

Query 26: Write a query in SQL to show the three minimum values of the integer column from the table.

Sol:

Syntax:

```
SELECT DISTINCT Column_Name FROM Table_Name a WHERE 3 <= (SELECT COUNT(DISTINCT Column_Name) FROM Table_Name b WHERE a. Column_Name <= b.Column_Name ) ORDER BY a. Column_Name DESC;
```

Example:

The following query shows the three minimum marks from the student table:

```
SELECT DISTINCT Stu_Marks FROM Student a WHERE 3 <= (SELECT COUNT(DISTINCT Stu_Marks) FROM Student b WHERE a. Stu_Marks <= b.Stu_Marks ) ORDER BY a.Stu_Marks DESC;
```

Query 27: Write a query to find the average of the integer column from the table.

Sol:

Syntax:

```
SELECT AVG (Column_Name) FROM Table_Name;
```

Example:

The following query finds the average of marks of Student table:

```
SELECT AVG (Stu_Marks ) FROM Student;
```

Query 28: Write a query to create a View in Structured Query Language,

Sol:

Syntax:

1. **CREATE VIEW** View_Name **AS SELECT** Column_Name1, Column_Name2, ... **FROM** Table_Name **WHERE** Condition;
2. **For** Creating a **View** in SQL, we have **to** use the **Create View** statement **with** the **SELECT** statement.

Example:

The following query creates the View of those students whose Marks is greater than 85 from the Student table:

1. **CREATE VIEW** Student_Age **AS SELECT** Stu_Name, Stu_Age **FROM** Student **WHERE** Stu_Marks < 85;

You can see the view table by using the following query:

1. **SELECT** * **FROM** Student_Age;

Output:

Chetan	20
Diksha	20
Sheetal	22

Query 29: Write a Query to add another column in the existing table:

Sol:

Syntax:

```
ALTER TABLE Table_Name ADD Column_Name Datatype ( Length_of_Column) ;
```

If you want to add another column or field to the existing table, you must use the ALTER statement in SQL.

Example:

The following query adds the Stu_Address column to the existing Student table:

```
ALTER TABLE Student ADD Stu_Addressvarchar (25) ;
```

Query 30: Write a query to transform any value into the specific SQL data type.

Sol:

The following query converts the floating-point value into the integer type.

```
SELECT CONVERT (int, 3025.58);
```